

# PyECLOUD

## Reference

## Manual

Giovanni Iadarola,  
Eleonora Belli, Philipp, Dijkstal,  
Lotta Mether, Annalisa Romano,  
Giovanni Rumolo, Eric Wulff  
CERN - Geneva, Switzerland

---

# PyECLOUD reference manual

This document describes the main input and output parameters for the PyECLOUD code for the simulation of the electron cloud buildup in particle accelerators.

## 1 Input files

### 1.1 Simulation parameters

This input file has to be named **simulation\_parameters.input**.

**Other input filenames.** The following variables specify the names of the other three input files which define the physical model of the simulation.

<b>machine_param_file</b>	Name of the machine parameter file.
<b>secondary_emission_parameters_file</b>	Name of the secondary emission configuration file.
<b>beam_parameters_file</b>	Name of the beam parameter file (in case of a multiple beam simulation this is the master beam).

**Secondary beams** The code can simulate the EC buildup in the presence of more than one circulating beam. For this purpose a list of secondary beam files (one for each beam) has to be provided. In the presence of secondary beams, the master beam determines the length of the simulation, the energy used for the calculation of the bending field, and the bunch spacing used for regeneration and saving purposes.

<b>secondary_beam_file_list</b>	(optional – default = []) List of names of secondary beam parameter files.
---------------------------------	---

#### Log and progress files

<b>logfile_path</b>	Path of a text file that reports some synthetic information on the ongoing simulation (Passage number, number of MPs and number of electrons in the chamber).
<b>progress_path</b>	Path of a text file that reports the simulation progress in percent.

**Time sampling** The simulated time interval is defined by the length of the beam profile (number of bunch passages) specified in the beam description.

<b>Dt</b>	[s] Simulation time step. This input is not considered if beam profile is imported from file.
-----------	---

<b>t_end</b>	[s] Extra time interval after the specified beam profile. Only coasting beam present in this interval. The value of this variable can also be negative in order to stop the simulation before the end of the specified profile.
<b>Negligible beam linear density</b>	
<b>lam_th</b>	[beam part/m] When the linear density of the beam is below this value, primary electrons are not generated and beam forces on the electrons are neglected.
<b>MP management settings</b>	
<b>N_mp_max</b>	Size of the arrays allocated to store the MP sizes, positions and velocities. MP regeneration settings must be set such that this number of MPs is never exceeded.
<b>N_mp_regen</b>	MP regeneration threshold. When the total number of MPs exceeds N_mp_regen a regeneration of the set of MPs is applied and a new Nref is chosen in order to get a target number of MPs. The check is performed at each $t=n*b\_spac$ with $n$ an integer.
<b>N_mp_regen_low</b>	Lower MP regeneration threshold. When the total number of MPs becomes smaller than N_mp_regen_low a regeneration of the set of MPs is applied and a new Nref is chosen in order to get a target number of MPs. The check is performed at each $t=n*b\_spac$ with $n$ an integer.
<b>t_ON_regen_low</b>	For $t < t\_ON\_regen\_low$ the lower MP regeneration check is not performed.
<b>N_mp_after_regen</b>	Target number of MPs obtained after regeneration.
<b>fact_split</b>	MP split factor; when secondary emission occurs, if the size of the emitted charge is smaller than $fact\_split*Nref$ , the impacting MP is simply rescaled according to the SEY. Otherwise new MPs are generated in order to keep the MP size as close as possible to Nref.
<b>fact_clean</b>	MP clean factor. MPs smaller than $fact\_split*Nref$ are deleted from the MP set. This test is performed at each $t=n*b\_spac$ with $n$ an integer.

<b>nel_mp_ref_0</b>	[e-/m] Initial MP size.
<b>Nx_regen, Ny_regen, Nvx_regen, Nvy_regen, Nvz_regen</b>	Number of mesh points in each dimension for the generation of the uniform grid for the 5-D phase space used for the MP regeneration.
<b>regen_hist_cut</b>	Defines a threshold value for the phase space density, below which no electrons are generated.
<b>N_mp_soft_regen</b>	(optional: by default soft regen is not applied, the presence of these inputs enables it) Soft regeneration threshold. The check is performed at each $t=n*b\_spac$ with $n$ an integer. <b>N_mp_soft_regen</b> should be smaller than <b>N_mp_regen</b> .
<b>N_mp_after_soft_regen</b>	(optional: by default soft regen is not applied, the presence of these inputs enables it) Target number of MPs obtained after a soft regeneration.

#### Space charge parameters

<b>Dt_sc</b>	[s] Time step for the update of the electron space charge field map.
<b>Dh_sc</b>	[m] Grid size for the space charge PIC solver.
<b>t_sc_ON</b>	[s] Electron space charge forces are neglected for $t < t\_sc\_ON$ – normally $t\_sc\_ON=0$ .
<b>sparse_solver</b>	[optional] choices are 'klu', 'scipy_slu' [default]
<b>PyPICmode</b>	[optional] Options are 'FiniteDifferences_ShortleyWeller' [default], 'ShortleyWeller_WithTelescopicGrids', 'FiniteDifferences_Staircase'

#### Multigrid parameters

To be used with **PyPICmode** = 'ShortleyWeller\_WithTelescopicGrids'

<b>f_telescope</b>	Magnification factor between grids (it must be always $0 < f < 1$ ).
<b>target_grid</b>	Target grid parameters. It is a dictionary containing: 'x_min_target', 'x_max_target', 'y_min_target', 'y_max_target' and 'Dh_target'.
<b>N_nodes_discard</b>	Number of nodes at the edge of the internal grids which are discarded in the field interpolation (going to coarser grid).
<b>N_min_Dh_main</b>	Minimum size of the first internal grid in the $\Delta h$ of the main grid.

Saving settings	
<b>filen_main_outp</b>	Main output filename or path. (optional - default 'Pyec1test.mat')
<b>dec_fact_out</b>	Output is saved only for every nth time step.
<b>Dx_hist</b>	[m] Defines the binning for the saving of the horizontal histograms.
<b>r_center</b>	[m] Radius of a circle around the (0, 0) point (center of the chamber) for the calculation of the local central density.
<b>Dt_En_hist</b>	[s] Time step used to store the energy spectrum of the electrons impacting on the chamber.
<b>Nbin_En_hist</b>	Number of bins used for the energy histogram.
<b>En_hist_max</b>	[eV] Maximum energy in energy spectrum. Larger energies are attributed to the last bin of the histogram.
<b>flag_cos_angle_hist</b>	Save the cosine of the incident angle of impacting electrons. (optional – default True)
<b>cos_angle_width</b>	Width of the histogram bins. (optional – default 0.05)
<b>flag_movie</b>	(optional – default=0) (1 $\Rightarrow$ On, 0 $\Rightarrow$ Off) Saves files with the electron density distribution at each space charge evaluation.
<b>flag_sc_movie</b>	(optional– default=0) (1 $\Rightarrow$ On, 0 $\Rightarrow$ Off) Saves files with the electron space charge electric field distribution at each space charge evaluation.
<b>save_mp_state_time_file</b>	(optional – default: no MP state is saved) [s] List of instants at which MP positions and velocities are saved on file. The list can also be provided as a .mat file.
<b>flag_detailed_MP_info</b>	(optional – default=0) (1 $\Rightarrow$ On, 0 $\Rightarrow$ Off) Enables save of MP number at each time step.
<b>flag_hist_impact_seg</b>	(optional – default=0) (1 $\Rightarrow$ On, 0 $\Rightarrow$ Off) If the chamber is a polygon, enables saving the number of electrons which impacts onto each segment of polygon.
<b>flag_verbose_file, flag_verbose_stdout</b>	(optional – default=False) True/False toggles to save detailed information of pathological impacts in output file and stdout respectively.

<b>dec_fac_secbeam_prof</b>	(optional – default=1) Decimation factor to decrease the number of points of the longitudinal beam profile of secondary beams.
<b>el_density_probes</b>	(optional – default: no probes) (List of Python dictionaries) Probes for electron density evaluation. For each of them it is necessary to indicate the position (x,y) and radius of the circle (r_obs).
<b>save_simulation_state_time_file</b>	(optional – default: no simulation state is saved) [s] List of instants at which the full simulation state is saved on file (pickle files). The list can also be provided as a .mat file.
<b>checkpoint_folder</b>	(optional – default: no checkpoints are saved) Path to folder where checkpoints are saved.
<b>checkpoint_DT</b>	(optional – default: no checkpoints are saved) Time interval in seconds between checkpoints being saved.
<b>copy_main_outp_folder</b>	(optional – default: output is not copied) Path to folder for storing backups of the output. Backups are needed to restart a simulation from a checkpoint.
<b>copy_main_outp_DT</b>	(optional – default: output is only copied if a checkpoint is saved) Enables backing up the output file in regular intervals.
<b>x_min_hist_det, x_max_hist_det, y_min_hist_det, y_max_hist_det, Dx_hist_det</b>	(optional – default: no detailed histogram is saved) [m] Defines a region of the chamber where a detailed horizontal distribution histogram is saved (_det stands for detailed).
<b>step_by_step_custom_observables, pass_by_pass_custom_observables, save_once_custom_observables</b>	(optional – default: no custom output) Custom observable definition. See example for detailed usage.
<b>Cloud particles</b>	
<b>cloud_mass</b>	(optional – default=electron mass) [kg] Mass of cloud particles.
<b>cloud_charge</b>	(optional – default=electron charge) [C] Charge of cloud particles.

**Additional clouds** Simulations with multiple clouds can be enabled with the following input parameter. See Section 1.5 for a detailed description of this simulation mode.

<b>additional_clouds_file_list</b>	(optional – default=[]) List of names of additional cloud files.
<b>SEY Extraction of the SEY curves can be disabled.</b>	
<b>extract_sey</b>	(optional – default=True) If set to False, it disables the extraction the SEY curves.
<b>Extraction of electron emission energy distribution</b>	
<b>extract_ene_dist</b>	(optional – default=False) If set to True, secondary emission energy distributions are extracted and saved.
<b>ene_dist_test_E_impact_eV</b>	The impacting energy used in the energy distribution extraction. [eV]
<b>D_ene_dist</b>	Defines the bin width for the extracted energy distribution. [eV]
<b>factor_ene_dist_max</b>	Defines the maximum energy included in the extracted energy distribution. The maximum energy is $\text{factor\_ene\_dist\_max} * \text{ene\_dist\_test\_E\_impact\_eV}$

## 1.2 Machine Parameters

Chamber profile description	
<b>chamb_type</b>	(optional – default = 'ellip') Possible settings: <ol style="list-style-type: none"><li>1. chamb_type = 'ellip' for elliptical chamber profile.</li><li>2. chamb_type = 'rect' for rectangular chamber profile.</li><li>3. chamb_type = 'polyg' or 'polyg_cython' for polygonal chamber profile.</li></ol>
When chamb_type = 'ellip' or 'rect' the following input variables must be provided:	
<b>x_aper, y_aper</b>	Horizontal and vertical semi-apertures of the transverse chamber profile.
When chamb_type = 'polyg' the following input variable must be provided:	
<b>filename_chm</b>	Name of file containing horizontal and vertical vertexes of the chamber profile.
<b>filename_chm_photoem</b>	Name of file containing horizontal and vertical vertexes of the chamber profile that is used for the photoemission mode 'per_segment'. The chamber file must also contain the cumulative distribution function of the emission probability for each segment. The vertexes of this chamber must lie on the edges of the main chamber.
<b>flag_assume_convex</b>	[optional] Default: True
<b>flag_counter_clockwise_chamb</b>	[optional] Default: True. Only needed for the photoemission model 'per_segment'. It specifies the order in which the vertexes are defined.
<b>Tracking and magnetic field</b> (the tracking algorithm has to be chosen according to the magnetic field conditions).	



<b>track_method</b>	(optional – default = ‘StrongBdip’) Possible settings:  1. track_method = ‘StrongBdip’ for vertical uniform magnetic field.  2. track_method = ‘StrongBgen’ for a generic transverse magnetic field map.  3. track_method = ‘BorisMultipole’ for arbitrary multipoles with Boris electron tracker.
---------------------	---

When track\_method = ‘StrongBdip’ the following input variables must be provided:

<b>B</b>	[T] Value of the vertical uniform magnetic field. If B=-1 the magnetic field is calculated from the total length of the bending magnets (bm_totlen) and from the beam energy.
<b>bm_totlen</b>	[m] Total length of dipoles inside the accelerator (it must be provided when B=-1).

When track\_method = ‘StrongBgen’ the following input variables must be provided:

<b>B_map_file</b>	Name of a .mat file containing the transverse field map. The case of a quadrupole magnet with unit gradient is embedded in the code and can be used setting: B_map_file = ‘analytic_quadrupole_unit_grad’.
<b>Bz_map_file</b>	Name of a .mat file containing the longitudinal field map. ??? The case of a quadrupole magnet with unit gradient is embedded in the code and can be used setting: B_map_file = ‘analytic_quadrupole_unit_grad’.
<b>fact_Bmap</b>	(optional – default = 1.) Scaling factor applied to the magnetic field map.
<b>B0x, B0y, B0z</b>	(optional – default = 0) [T] Uniform magnetic fields added to the map.
<b>B_zero_thrld</b>	[T] Value below which the local magnetic field is approximated with 0.

When track\_method = ‘BorisMultipole’, N\_sub\_steps and B\_multip must be provided, while B\_skew is optional:

<b>N_sub_steps</b>	Number of tracking sub-steps per time step.
<b>B_multip</b>	Magnetic momenta. Higher order multipoles and skew magnets are supported, as well as compound magnets. The formalism in PyE-CLOUD follows this formula:

$$B_y + iB_x = \sum_{n=0}^{\infty} \frac{1}{n!} (b_n + ib'_n) (x + iy)^n$$

	B_multip specifies $b_n = \frac{\partial^n B_y}{\partial x^n}$ . For example: to simulate a dipole with B=8.3 T set B_multip = [8.3]; to simulate a sextupole with 100 T/m <sup>2</sup> set B_multip = [0., 0., 100.].
<b>B_skew</b>	The skew parameters can be included through B_skew. For a skew quadrupole with a gradient of 12 T/m, set B_multip to [0.,0.] and B_skew to [0.,12.]. B_skew specifies $b'_n = \frac{\partial^n B_x}{\partial x^n}$ and defaults to <i>None</i> .

**Optics parameters** (can be provided also in the beam parameter file(s) independently for the different beams) — not needed if transverse beam size is directly defined in the beam parameter files.

<b>betafx, betafy</b>	[m] Horizontal and vertical beta functions at the simulation section.
<b>Dx, Dy</b>	(optional – default = 0) [m] Horizontal and vertical dispersion functions at the simulation section.

**Residual gas ionization parameters** (if the following input parameters are omitted primary electron generation by residual gas ionization is not enabled).

<b>gas_ion_flag</b>	(optional – default=0) (1 $\Rightarrow$ On, 0 $\Rightarrow$ Off) Enables primary electron generation by residual gas ionization.
<b>P_nTorr</b>	[nTorr] Pressure in the vacuum chamber.
<b>sigma_ion_MBarn</b>	[MBarn] Ionization cross section of the residual gas.
<b>Temp_K [K]</b>	Temperature of the residual gas.
<b>unif_frac</b>	Fraction of primary electrons uniformly distributed inside the chamber. The remaining part is generated according to the transverse distribution of the beam.
<b>E_init_ion</b>	[eV] Initial energy of the generated electrons.
<b>t_ion</b>	???

**Photoemission parameters** (if generation by photoemission is not desired, the following parameters can be omitted).

<b>photoem_flag</b>	(optional – default=0) (0 $\Rightarrow$ Off, 1 $\Rightarrow$ traditional, 2 or 'from_file' $\Rightarrow$ Angular distribution from file, 3 or 'per_segment' $\Rightarrow$ Part of chamber definition) Enables primary electron generation by photoemission.
<b>k_pe_st</b>	[m <sup>-1</sup> ] Number of photoelectrons to be generated per proton and per unit length.
<b>energy_distribution</b>	<p>Can be one of:</p> <ul style="list-style-type: none"> <li>• 'gaussian': <math>p(E) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(E-\mu)^2}{2\sigma^2}\right)</math></li> <li>• 'log-normal': <math>p(E) = \frac{1}{E\sigma\sqrt{2\pi}} e^{-\frac{(\log(E)-\mu)^2}{2\sigma^2}}</math></li> <li>• 'rect': <math>p(E)</math> uniform for <math>\mu - \sigma &lt; E &lt; \mu + \sigma</math></li> <li>• 'mono': <math>E = \mu</math></li> <li>• 'lorentz': <math>p(E) = \frac{1}{\pi} \frac{\sigma}{\sigma^2 + (E-\mu)^2}</math></li> </ul> <p>The Lorentz and normal distributions are cut at 0 to prevent negative energies of photoemitted electrons.</p>
<b>e_pe_sigma</b>	[eV] $\sigma$ from above.
<b>e_pe_max</b>	[eV] $\mu$ from above.
<b>out_radius</b>	[m] Radius of a circle external to the chamber.
<b>phem_resc_fact</b>	Rescaling factor on the position vector of the generated photoelectrons.
<b>photoelectron_angle_distribution</b>	See secondary_angle_distribution in the secondary emission parameters.
<b>Photoemission parameters for photoem_flag = 1</b> <b>A part of photoelectrons is generated in the corner of the beam screen on direct impact of photons (non-reflected). The reflected photons generated elsewhere.</b>	
<b>inv_CDF_refl_photoem_file</b>	Name of a .mat file providing the inverse of the Cumulative Distribution Function (CDF) for the angular distribution of the reflected photons. If inv_CDF_refl_photoem_file = 'unif_no_file' the reflected photons have uniform angular distribution and no file needs to be provided.
<b>refl_frac</b>	Fraction of photoelectrons generated by reflected photons.
<b>x0_refl, y0_refl</b>	[m] Impact of non-reflected photons.
<b>alimit</b>	[rad] Extent ( $1\sigma$ ) of the Gaussian angular distribution of photoelectrons generated by non-reflected photons.

**Photoemission parameters for photoem\_flag = 2 or 'from\_file'**

**The coordinates of all generated photoelectrons is specified from a file**

<b>inv_CDF_all_photoem_file</b>	See inv_CDF_refl_photoem_file, but for all photons. If it is set to 'unif_no_file', the photoelectron generation is uniform over the whole surface. Example scripts to generate a file with the correct format can be found in the folder PyECLoud/other/photoemission_angular_distribution.
---------------------------------	--

**Uniform initial distribution** Simulation starts with electrons uniformly distributed in the chamber (if the following input parameters are omitted this feature is not enabled).

<b>init_unif_flag</b>	(optional – default=0) (1 $\Rightarrow$ On, 0 $\Rightarrow$ Off) Enables uniform electron distribution at the beginning of the simulation.
<b>Nel_init_unif</b>	Initial number of electrons.
<b>E_init_unif</b>	[eV] Initial energy of the generated electrons.
<b>x_max_init_unif, x_min_init_unif, y_max_init_unif, y_min_init_unif</b>	[m] Edges of the region where the electrons are generated.

### 1.3 Beam parameters

Basic definitions	
<b>m0_part</b>	(optional – default=proton mass) [Kg] Mass of beam particles.
<b>q_part</b>	(optional – default=proton charge) [C] Charge of beam particles.
<b>energy_eV</b>	[eV] Total energy of the beam particles.
<b>Dp_p</b>	(optional – default=0) Momentum spread (r.m.s.) of the beam. This parameter is not used if the transverse beam size is directly provided (vars. sigmax, sigmay).
<b>nemittx, nemitty</b>	[m] Normalized transverse emittance of the beam. These parameters are not used if the transverse beam size is directly provided (vars. sigmax, sigmay).
<b>x_beam_pos, y_beam_pos</b>	(optional – default=0) [m] Transverse position of the beam within the vacuum chamber.
<b>sigmax, sigmay</b>	(optional – default=-1) [m] Transverse geometrical beam size. If sigmax, sigmay=-1 the beam size is calculated through the energy, the normalized emittance, the dispersion and the momentum spread of the beam.
Transverse electric field of the beam	
<b>beam_field_file</b>	Name of a .mat file containing the map of the transverse beam electric field. If beam_field_file = -1 or beam_field_file = 'computeFD' the beam field map is computed using the embedded Finite Difference (FD) Poisson solver. If beam_field_file='computeBE' the electric field is computed using the Bassetti-Erskine formula and the image terms for the elliptical boundary conditions (this option is available only when the chamber profile is elliptical). If beam_field_file='compute_FDSW_multigrid' the beam field map is computed using the embedded Finite Difference (FD) Poisson solver in the multigrid case.
<b>save_beam_field_file_as</b>	(optional – default: the beam field map is not saved) Name of the file where the employed field map is saved.

When beam_field_file = -1 or beam_field_file = 'computeFD' the following parameter must be provided:	
<b>Dh_beam_field</b>	Grid size for the FD solver and for the saved field map.
When beam_field_file = 'computeBE' the following parameters must be provided:	
<b>Nx, Ny</b>	Number of points of the employed field map.
<b>nimag</b>	Number of image terms.
When beam_field_file = 'compute_FDSW_multigrid' the following parameters must be provided:	
<b>Dh_beam_field</b>	Grid size for the FD solver and for the saved field map.
<b>f_telescope_beam</b>	Magnification factor between grids (it must be always $0 < f < 1$ )
<b>target_grid_beam</b>	Target grid parameters. It is a dictionary containing: 'x_min_target', 'x_max_target', 'y_min_target', 'y_max_target' and 'Dh_target'.
<b>N_nodes_discard_beam</b>	Number of nodes at the edge of the internal grids which are discarded in the field interpolation (going to coarser grid).
<b>N_min_Dh_main_beam</b>	Minimum size of the first internal grid in the $\Delta h$ of the main grid.
<b>Beam longitudinal profile</b>	
<b>b_spac</b>	Bunch spacing. It is used to generate the beam profile when it is defined in the form of a bunched beam. It is also the time interval for cleaning and regeneration of the MP set, as well as for saving.
<b>fact_beam</b>	Rescaling factor applied to the beam profile.
<b>coast_dens</b>	[p/m] Coasting beam density added to the beam profile.
<b>flag_bunched_beam</b>	Two possible settings: <ol style="list-style-type: none"> <li>1. flag_bunched_beam = 0 the beam profile is loaded from file.</li> <li>2. flag_bunched_beam = 1 the beam profile is generated from a filling pattern.</li> </ol>
When flag_bunched_beam = 1 the following parameters must be provided:	
<b>sigmaz</b>	[m] Bunch length ( $1\sigma$ ).

<b>t_offs</b>	[s] Delay in the longitudinal profile (n.b. if t_offs=0 only half of the first bunch is simulated).
<b>filling_pattern_file</b>	Name of a file providing the intensities of the different bunches (zeros for empty slots). The data can be provided also in form of a python list with no need to provide an input file, e.g. filling_pattern_file=4*(72*[1.]+8*[0.]).
When flag_bunched_beam = 0 the following parameter must be provided:	
<b>beam_long_prof_file</b>	Name of a .mat file providing the longitudinal beam profile. This file will also define the time step used for the simulation.

## 1.4 Secondary emission parameters

Choice of the model	
<b>switch_model</b>	(optional – default=0) Different secondary emission models: <ol style="list-style-type: none"> <li>1. switch_model = 0 or 'ECLOUD' for the model used in ECLOUD.</li> <li>2. switch_model = 'ACC_LOW' for a more accurate treatment of low energy impacts.</li> <li>3. switch_model = 'ECLOUD_nunif' for the model used in ECLOUD, with sey info in the chamber shape file.</li> <li>4. switch_model = 'ECLOUD_nunif_charging' for the model used in ECLOUD and charging effects, with sey and charging info in the chamber shape file.</li> <li>5. switch_model = 'cos_low_ene' for cosine low energy dependence.</li> <li>6. switch_model = 'flat_low_ene' for flat low energy dependence.</li> <li>7. switch_model = 'from_file' for interpolation of a user-specified curve.</li> </ol>
Secondary Electron Yield	
These are the parameters for the 'ECLOUD' model, as described in G. Iadarola's thesis.	
<b>E<sub>max</sub></b>	Energy corresponding to the maximum SEY.
<b>s_param</b>	Shape parameter of the true SEY curve.
<b>del_max</b>	Maximum of the SEY curve.
<b>R0</b>	Weight of the reflected electron component.
<b>E0</b>	Shape parameter of the reflected SEY curve.
<b>E<sub>th</sub></b>	Maximum energy for true secondary electrons.
<b>sigmafit</b>	Sigma parameter of the lognormal distribution.
<b>mufit</b>	Mu parameter of the lognormal distribution.
Other parameters	



<b>sey_file</b>	Needed for secondary emission model 'from_file'. Path to a file that specifies the reflected and true secondary emission yields for different energies. An example can be found in the subfolder sey_files in the PyECLOUD code.
<b>flag_costheta_delta_scale</b>	If enabled the SEY curve associated to true secondary electrons is scaled as a function of the angle of the impacting electron. Default is TRUE.
<b>flag_costheta_Emax_shift</b>	If enabled the SEY curve associated to true secondary electrons is shifted (change of energy scale) as a function of the angle of the impacting electron. Default is TRUE.
<b>switch_no_increase_energy</b>	switch_no_increase_energy = 1 checks that the true secondaries do not have more energy than the corresponding impacting electrons. Typically this feature is NOT enabled.
<b>thresh_low_energy</b>	Maximum energy for which the energy check is performed.
<b>scrub_en_th</b>	Minimum energy of scrubbing electrons (for scrubbing current estimations).
<b>secondary_angle_distribution</b>	Can be 'cosine_2D', 'cosine_3D' or 'normal_emission'. For new electrons, $\theta$ describes the angle between the surface normal and their initial velocity vector. $dn/d\theta = \cos\theta$ or $dn/d\theta = \cos\theta \sin\theta$ in the 2D or 3D cases, respectively. More info here. The more accurate is the 3D cosine distribution which takes into account the surface element $\sin\theta$ in spherical coordinates. For 'normal_emission', $\theta = 0$ .

## 1.5 Simulations with multiple clouds

The code provides the option of including more than one cloud in the simulation. Each cloud is influenced by the other clouds only through their space charge. This option is switched on by providing a list of names of additional cloud files in the standard input, as follows.

In the **simulation\_parameters.input** file:

<b>additional_clouds_file_list</b>	(optional – default=[]) List of names of additional cloud files.
------------------------------------	---

Each cloud file should contain a subset of the input parameters specified in the main input files, as listed below. The following parameters are required in the cloud input file.

### Mandatory cloud parameters

<b>cloud_mass</b>	See Simulation parameters section.
<b>cloud_mass</b>	
<b>gas_ion_flag</b>	See Machine parameters section.
<b>photoem_flag</b>	
<b>init_unif_flag</b>	
<b>init_unif_edens_flag</b>	
<b>switch_model</b>	See Secondary emission parameters section.

In addition, a cloud file can contain the following optional parameters. Any optional parameter that is not specified in the cloud input file takes its value from the main input, and gets the default value if the parameter is not given in the main input files.

### Optional cloud parameters

<b>MP management settings</b>	See the Simulation parameters section for the detailed list of parameters and their usage.
<b>Saving settings</b>	
<b>Log and progress files</b>	
<b>N_sub_steps</b>	See under “Tracking and magnetic field” in the Machine parameters section.
<b>Residual gas ionization parameters</b>	See the Machine parameters section for the detailed list of parameters and their usage.
<b>Photoemission parameters</b>	
<b>Uniform initial distribution</b>	
<b>Uniform initial density</b>	
<b>Secondary emission parameters</b>	As listed in the Secondary emission parameters section.

Each cloud produces its own .mat output file, as described in the following chapter. The file is named according to the main output file name and the cloud file name.

## 2 Output

The main PyECLOUD output is in a single .mat file (by default called 'Pyecldtest.mat'). It can be opened in python, for example using the myloadmat\_to\_obj module in the main PyECLOUD folder, or using the scipy.io module.

The file contains arrays of different shape. The time steps in the simulation are spaced by  $\Delta t$  from the simulation parameters input file. However the output is only saved every  $n$ th time step, where  $n$  is specified by dec\_fact\_out. The number of these steps is denoted  $N_{\text{step}}$  in the following.

Variables saved at every time step		
t( $N_{\text{steps}}$ )	[s]	The time at all time steps where the output is saved.
En_emit_eV_time( $N_{\text{steps}}$ )	[eV]	Combined energy of all electrons emitted since the last step.
En_imp_eV_time( $N_{\text{steps}}$ )	[eV]	Combined energy of all electrons impacted since the last step.
En_kin_eV_time( $N_{\text{steps}}$ )	[eV]	Combined kinetic energy of all electrons currently in the chamber.
Nel_emit_time( $N_{\text{steps}}$ )		Number of electrons emitted since the last step.
Nel_imp_time( $N_{\text{steps}}$ )		Number of electrons impacted since the last step.
Nel_timep( $N_{\text{steps}}$ )		Number of electrons currently in the chamber.
cen_density( $N_{\text{steps}}$ )	[m <sup>-3</sup> ]	Electron density in a sphere around the center of the chamber with radius r_center.
lam_t_array( $N_{\text{steps}}$ )	[m <sup>-1</sup> ]	Current line density of the beam charge.

$N_{\text{pass}}$  is defined as the number of bunch passages.  $N_{\text{bins}}$  is the number of bins in the horizontal plane used to produce related histograms.  $N_{\text{en}}$  is the number of bins in impact energy used to produce related histograms.

Variables saved at each bunch passage		
t_hist( $N_{\text{pass}}$ )	[s]	The time at which data for the bunch passages are saved.
En_hist		Energy spectrum of the impacting electrons (electron per time interval and per energy bin). The last bin contains all electrons falling outside the selected energy interval.
all_Ekin_hist		Kinetic energy histogram of all the electrons in the chamber (electron per time interval and per energy bin). The last bin contains all electrons falling outside the selected energy interval.
t_En_hist	[s]	The time at which the electron energy spectrum is saved.

En_g_hist	[eV]	Center of the energy bins used to save the energy spectrum.
xg_hist(N_bins)	[m]	The $x$ coordinates corresponding to related histogram bins.
N_mp_corrected_pass(N_pass)		Number of times a MP tracking error occurred.
N_mp_impact_pass(N_pass)		Number of MPs that impacted.
N_mp_pass(N_pass)		Number of current MPs.
N_mp_ref_pass(N_pass)		Current reference MP size.
energ_eV_impact_hist(N_pass, N_bins )		Horizontal histogram of the energy deposited on the walls by the electrons.
nel_hist(N_pass, N_bins )		Horizontal histogram of the electron positions.
nel_impact_hist_scrub(N_pass, N_bins )		Horizontal histogram of the electron impacts with an energy higher than scrub_en_th.
nel_impact_hist_tot(N_pass, N_bins )		Horizontal histogram of all electron impacts.

#### Other variables

dec_fact_out		Timestep output is only saved every nth time step, where n corresponds to dec_fact_out.
b_spac	[s]	Bunch spacing used for simulation internals.
t_sc_video	[s]	The times at which the space charge forces are recomputed.
U_sc_eV		Not used.
N_mp_time		Only used when flag_detailed_MP_info. Save the total number of macroparticles at each time step.
nel_hist_impact_seg		Number of electrons that impacted on each chamber segment. Only used if flag_hist_impact_seg is set.
energ_eV_impact_seg	[eV]	Combined energy of all impacted electrons on each chamber segment. Only used if flag_hist_impact_seg is set.
t_sec_beams	[s]	Time where information is stored on secondary beams.
sec_beam_profiles [m <sup>-1</sup> ]		Charge profile of secondary beams.
el_dens_at_probes	[m <sup>-3</sup> ]	Electron densities at locations specified with el_density_probes.
x_el_dens_probes	[m]	$x$ coordinates of electron density probes
y_el_dens_probes	[m]	$y$ coordinates of electron density probes
r_el_dens_probes	[m]	Radii used to measure electron density probes

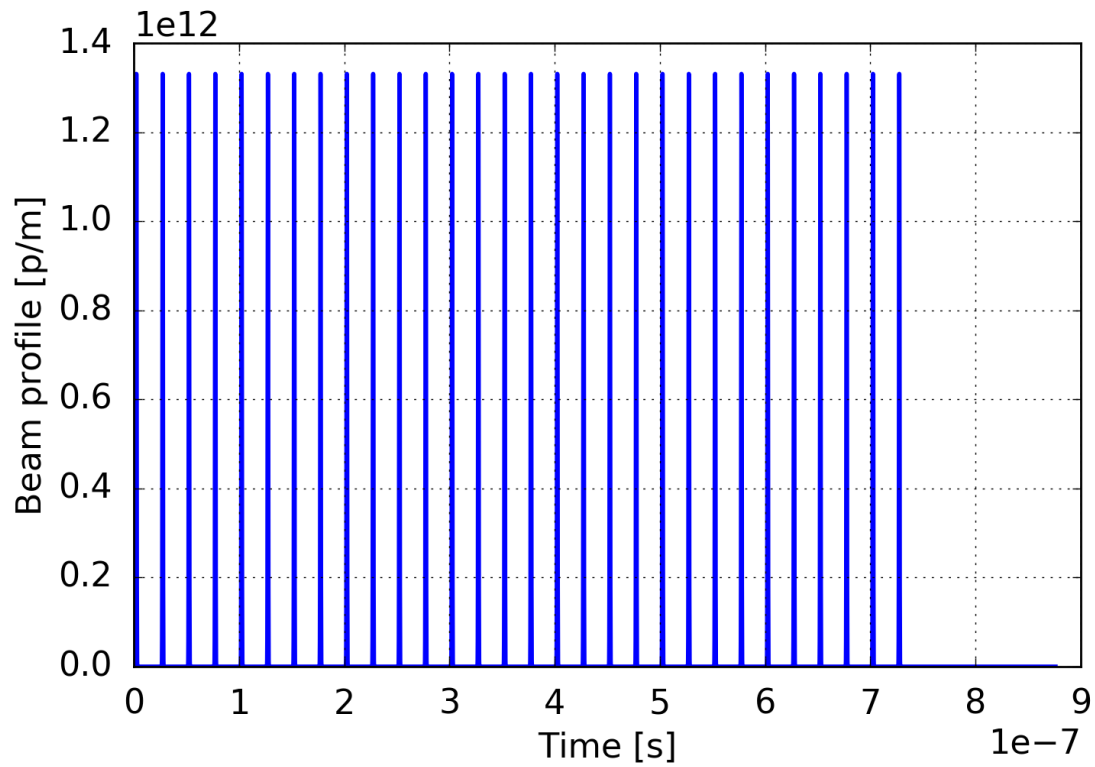
nel_hist_det	Detailed electron histogram. Only used if flag_hist_det is set.
nel_hist_det	Coordinates corresponding to the bins of nel_hist_det. Only used if flag_hist_det is set.
cos_angle_hist	The cosine of the impact angles of all electrons.
xg_hist_cos_angle	The bins corresponding to the cos_angle_hist.

Example plots, for one of the simulations in the PyECLOUD test suite, are produced by the script:

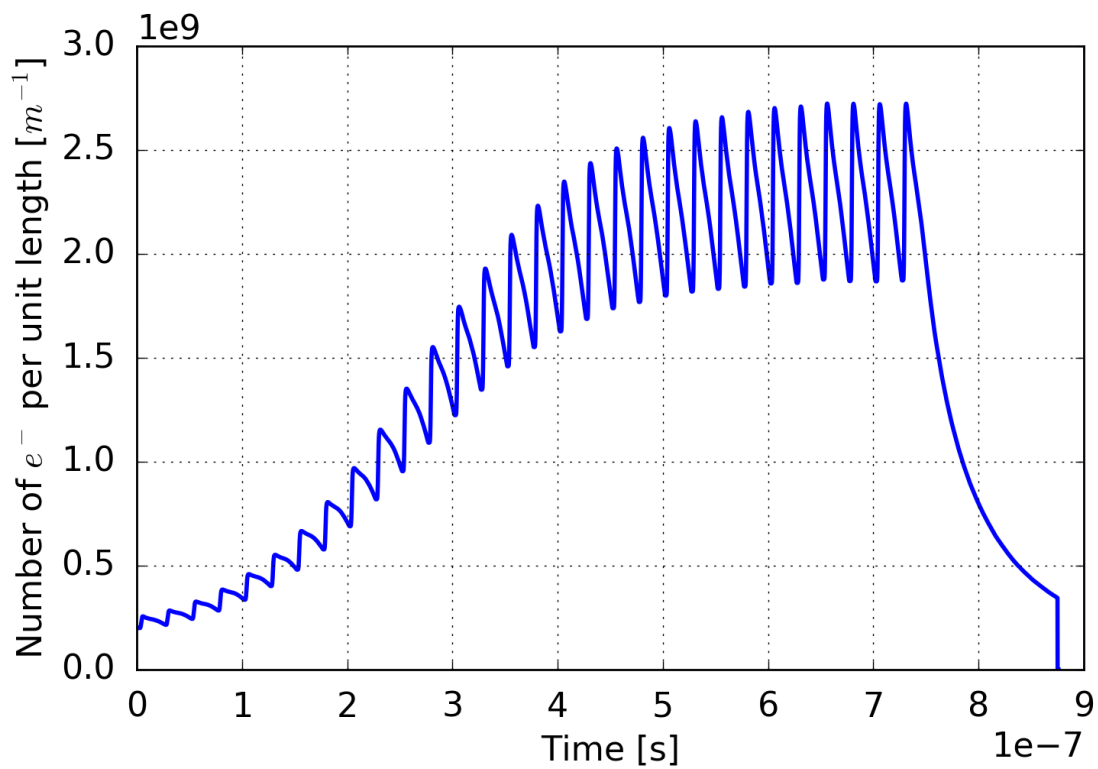
doc/example/000\_plot\_main\_output.py

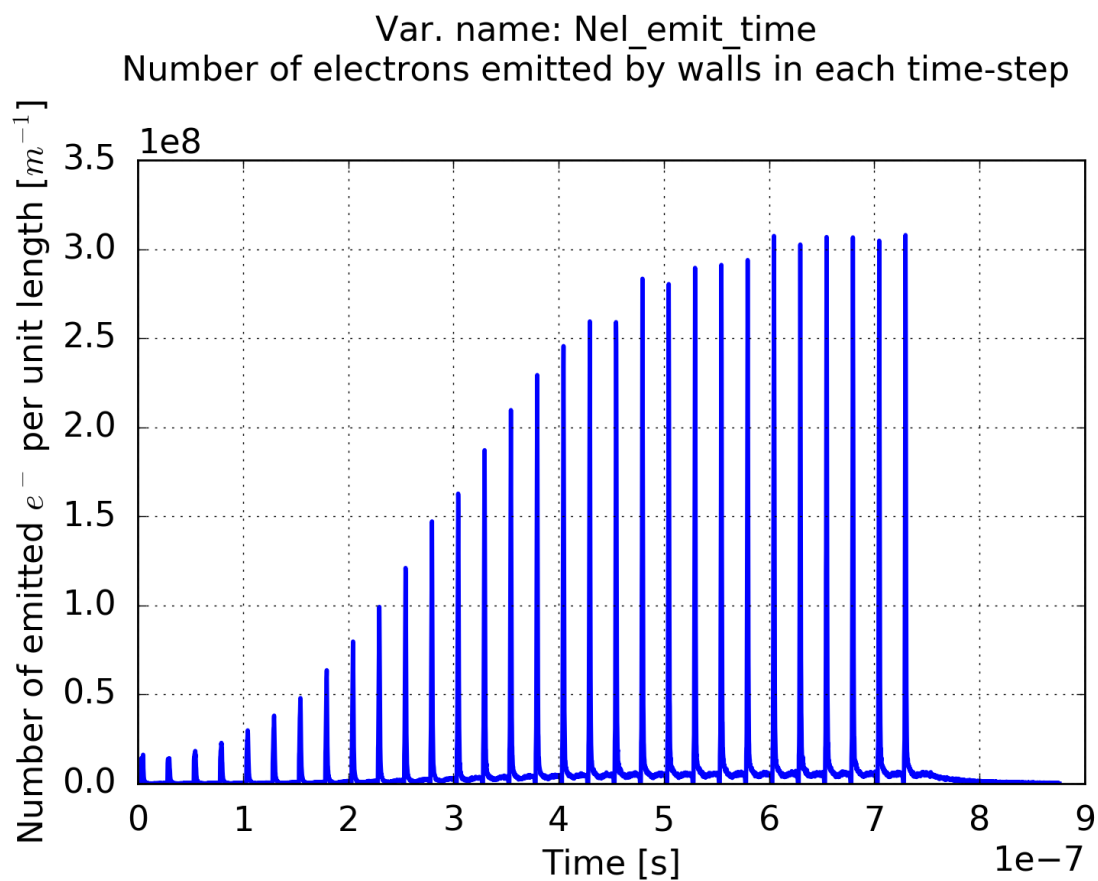
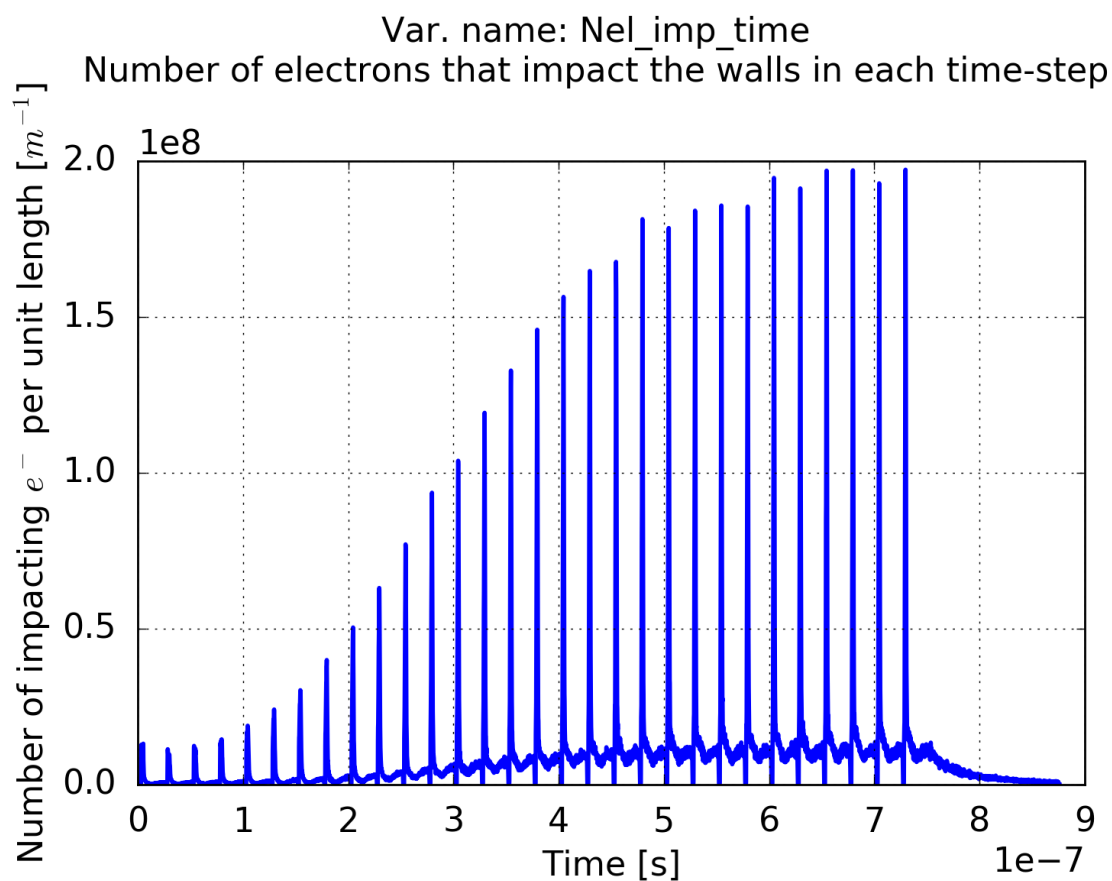
and reported in the following pages (with some description).

Var. name: lam\_t\_array  
Beam density at each time step

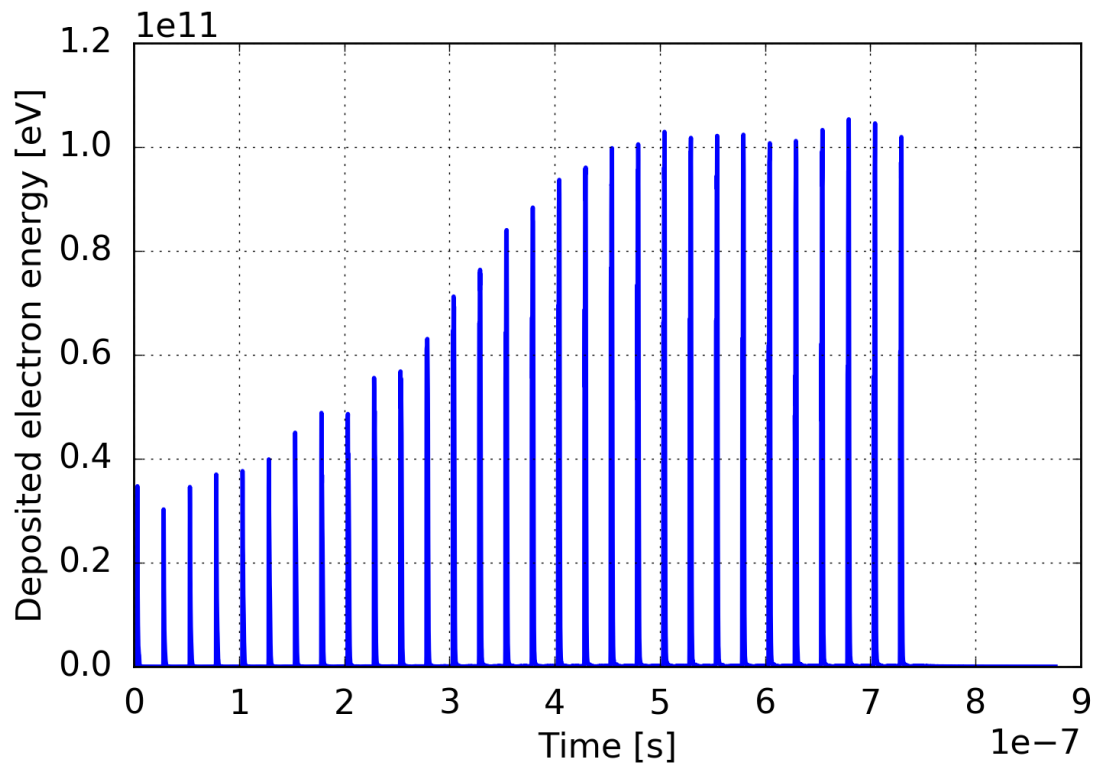


Var. name: Nel\_timep  
Number of electrons in the chamber at each time step

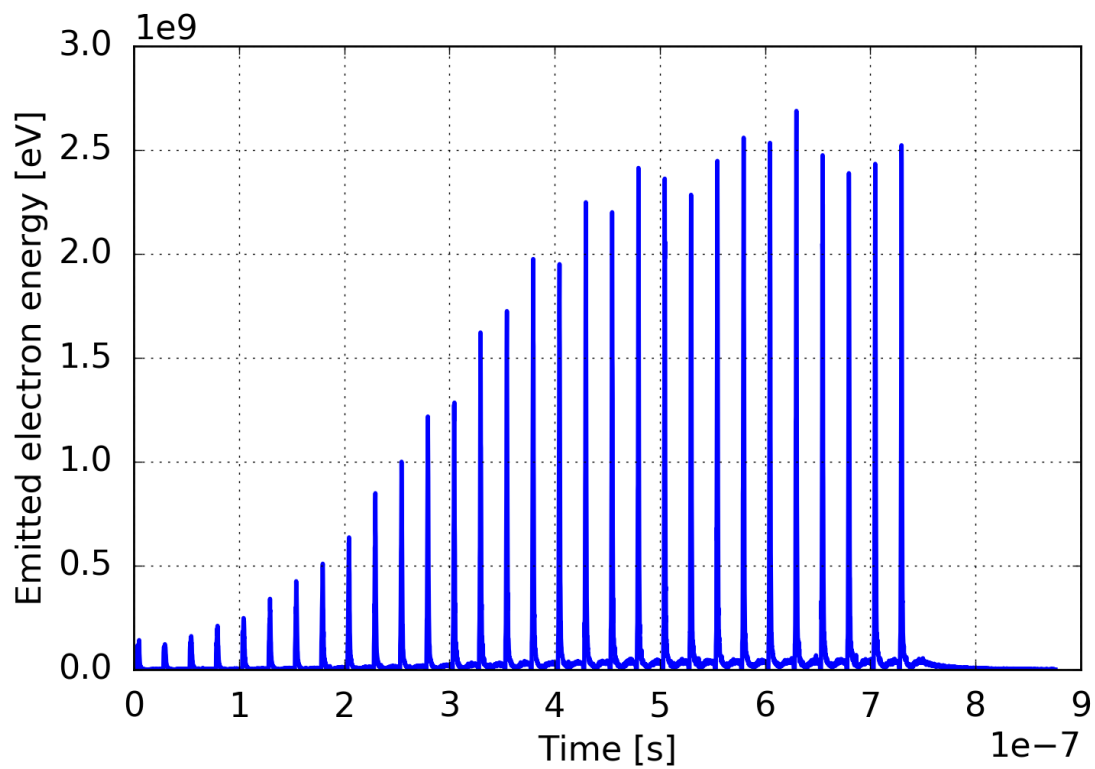




Var. name: En\_imp\_eV\_time  
Electron energy deposited on the walls in each time-step

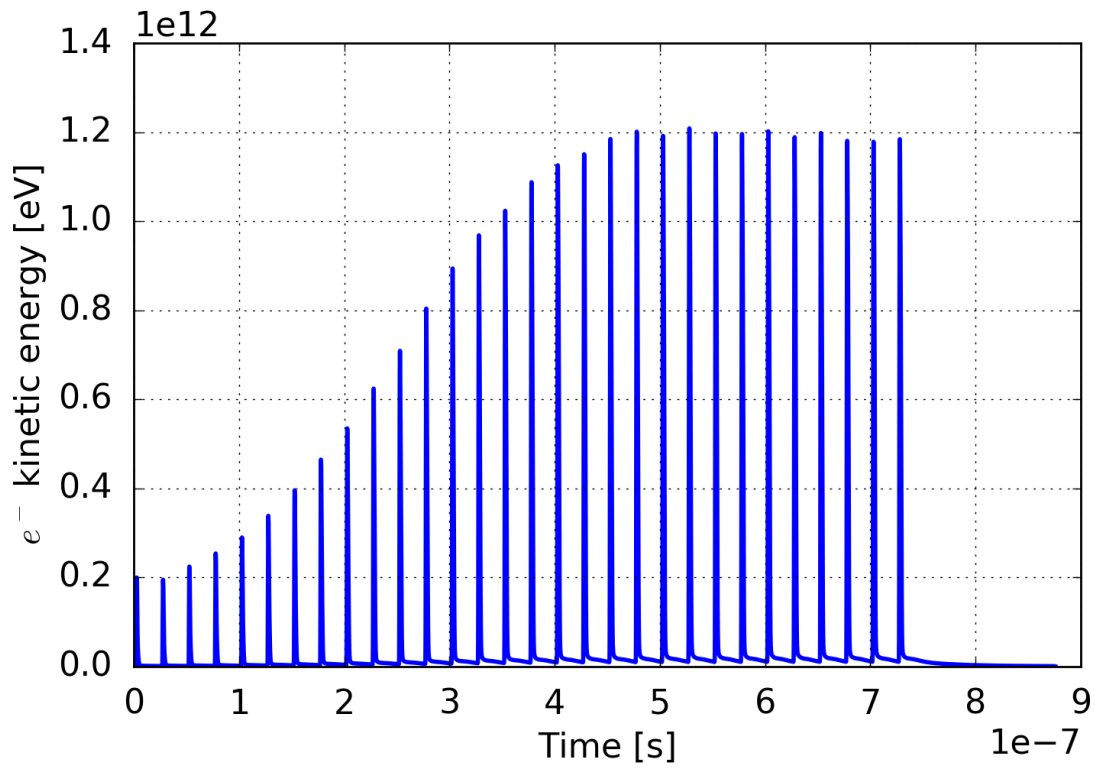


Var. name: En\_emit\_eV\_time  
Electron energy emitted by the walls in each time-step

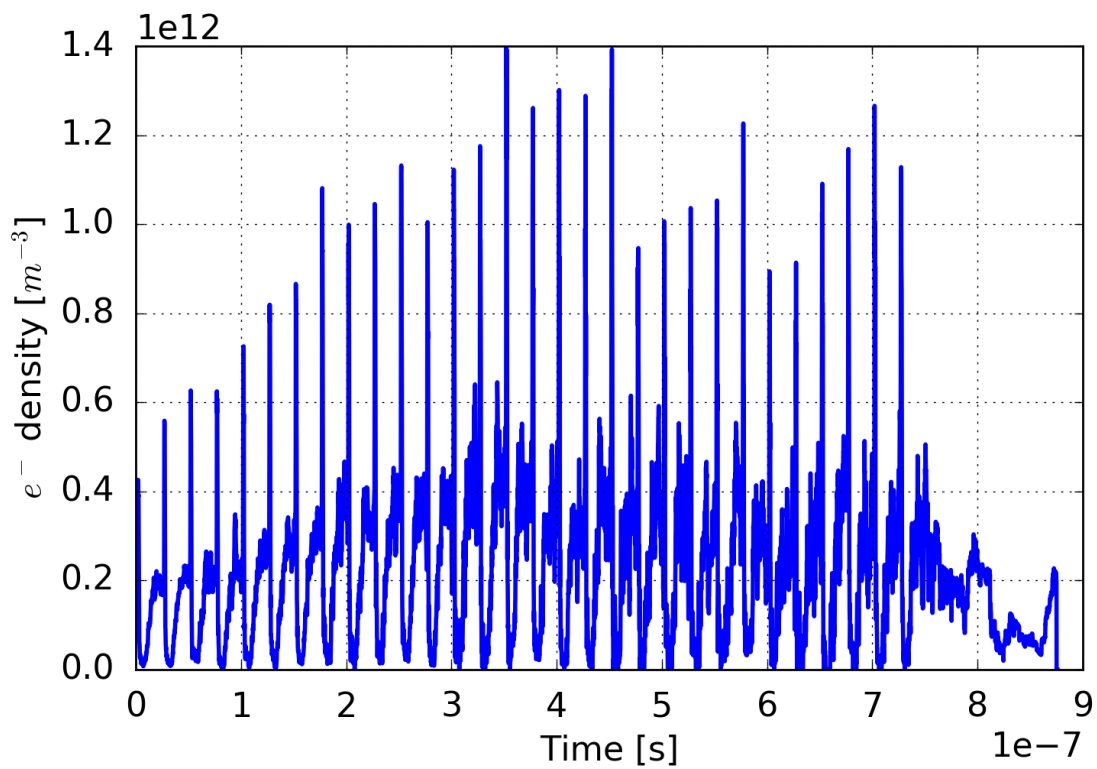




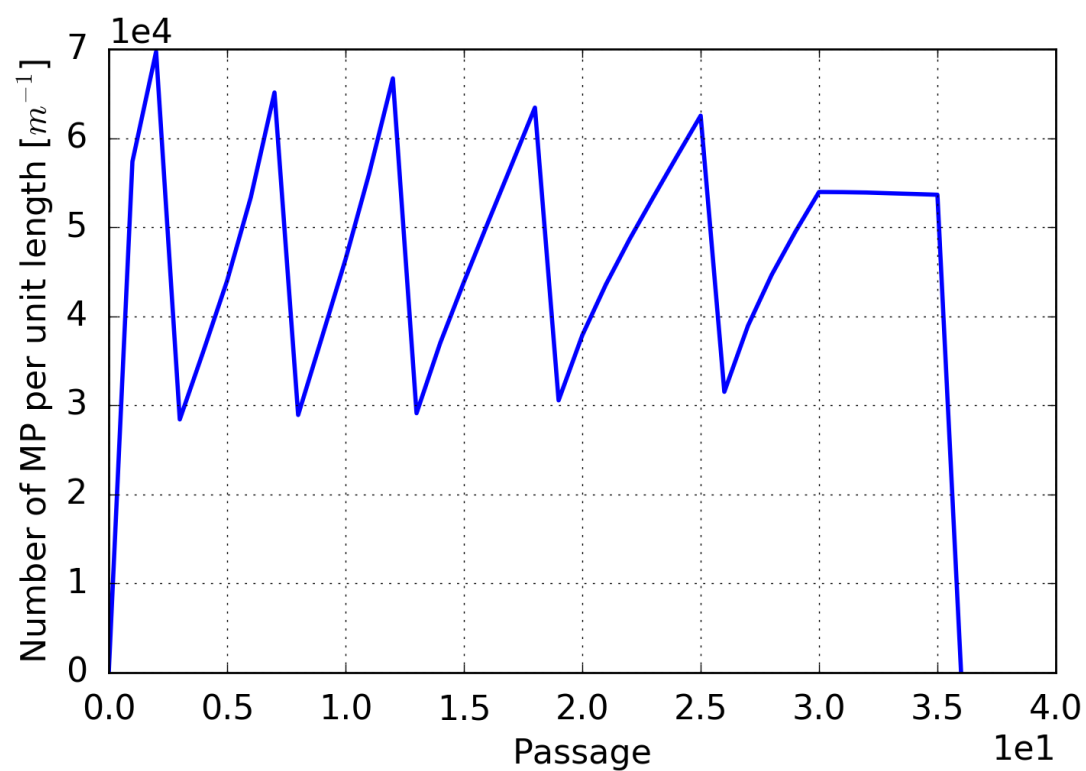
Var. name: En\_kin\_eV\_time  
Total kinetic energy of the electrons at each time-step



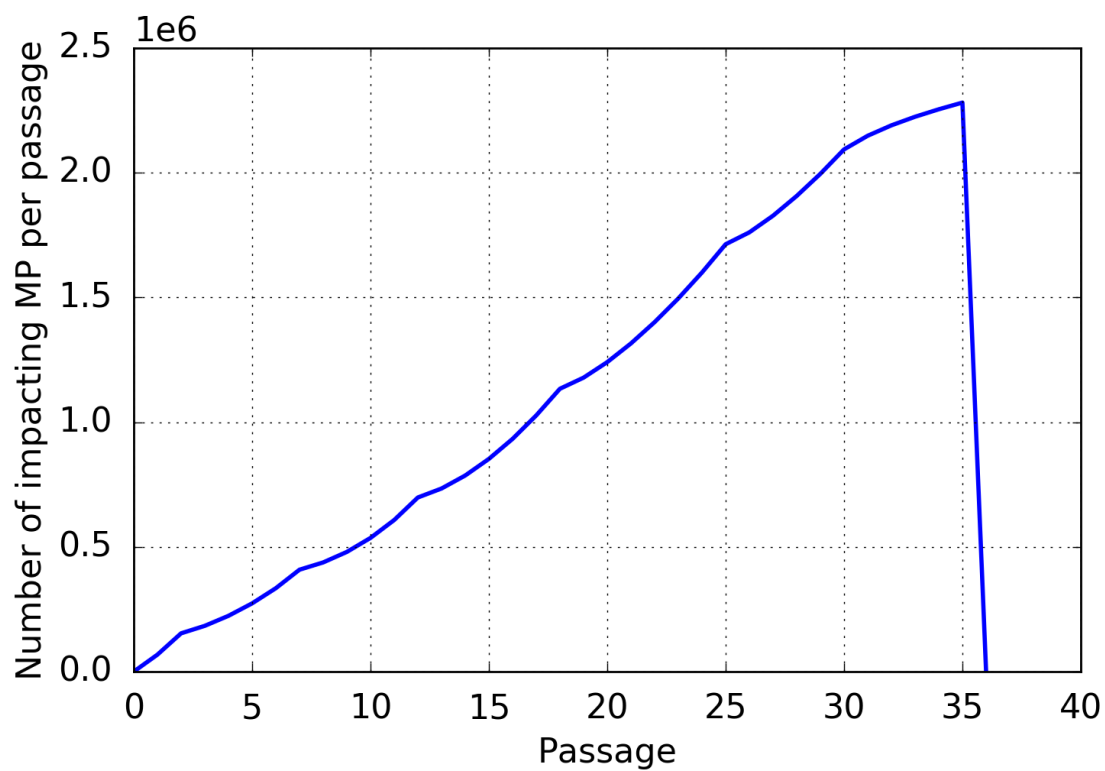
Var. name: cen\_density  
electron density at the beam position



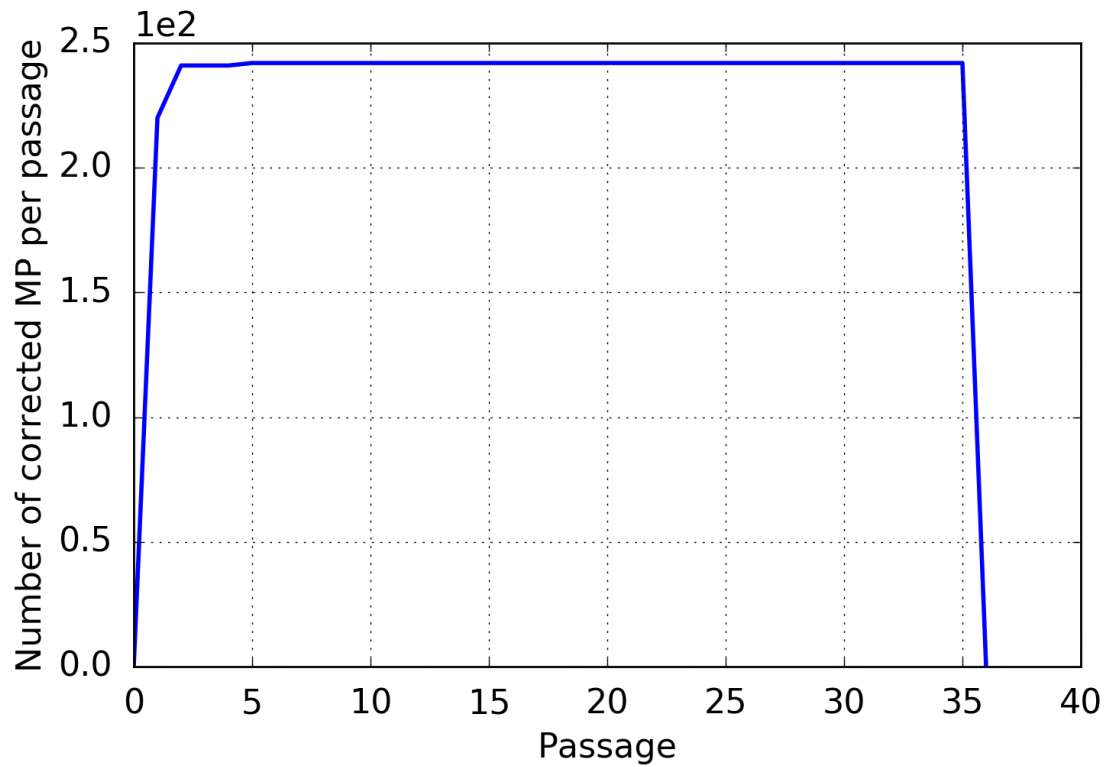
Var. name: N\_mp\_pass  
Number of MP at each passage



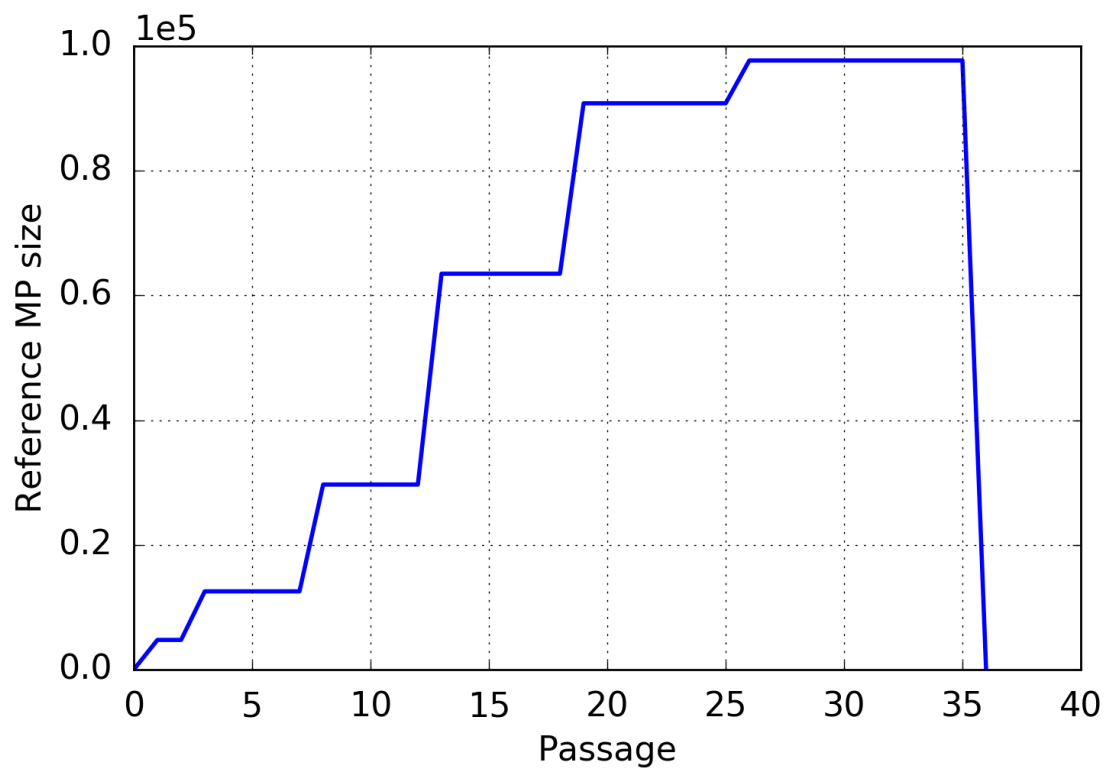
Var. name: N\_mp\_impact\_pass  
Number of macroparticles that impact for each passage



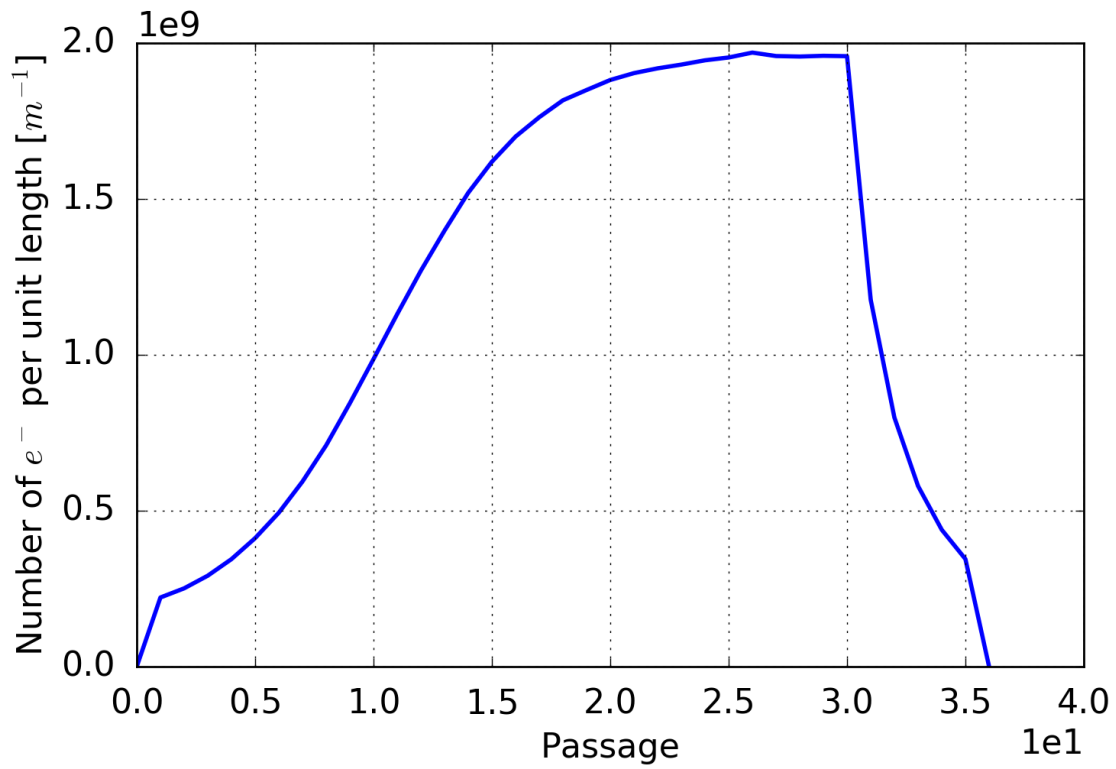
Var. name: N\_mp\_corrected\_pass  
Number of macroparticles for which fallback algorithm is used.



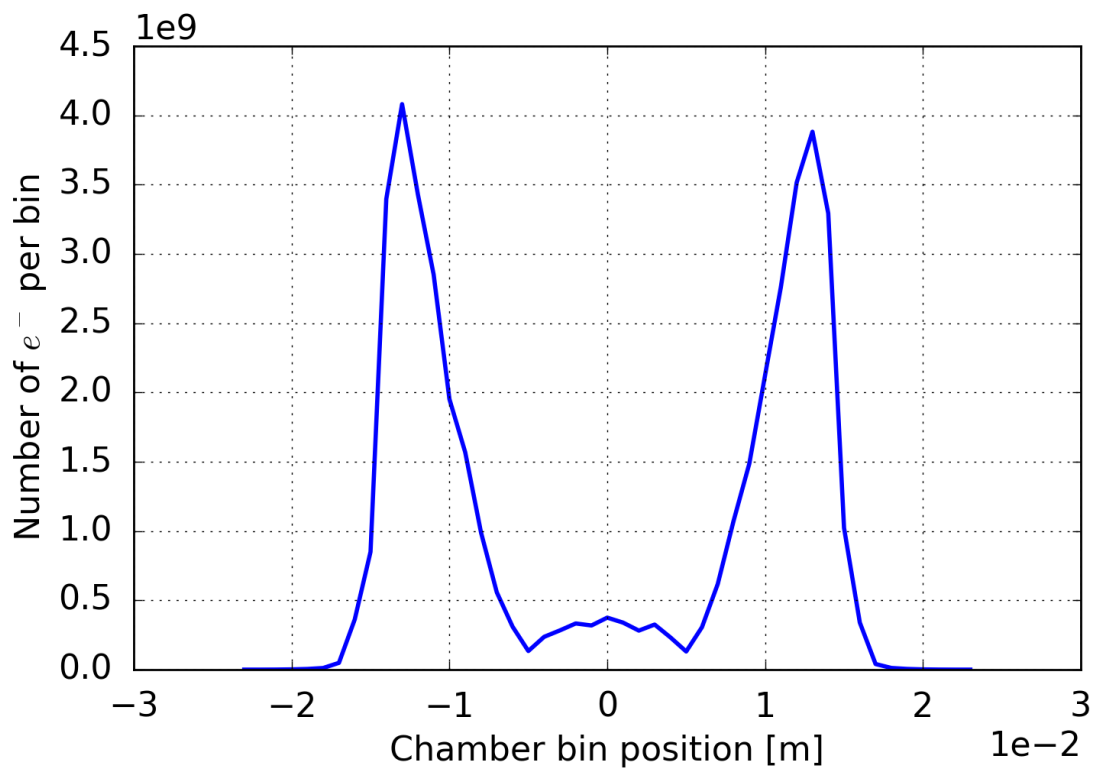
Var. name: N\_mp\_ref\_pass  
Reference macroparticle size at each passage



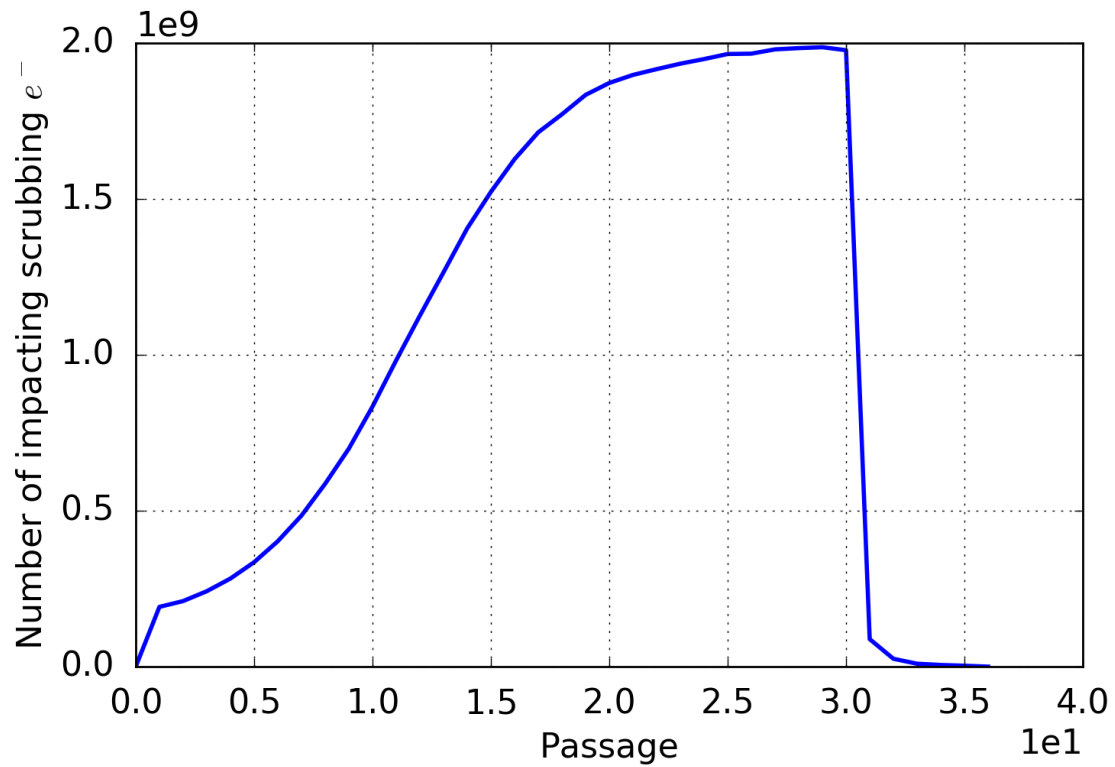
Var. name: sum(nel\_hist, axis=1)  
Number of electrons at each passage



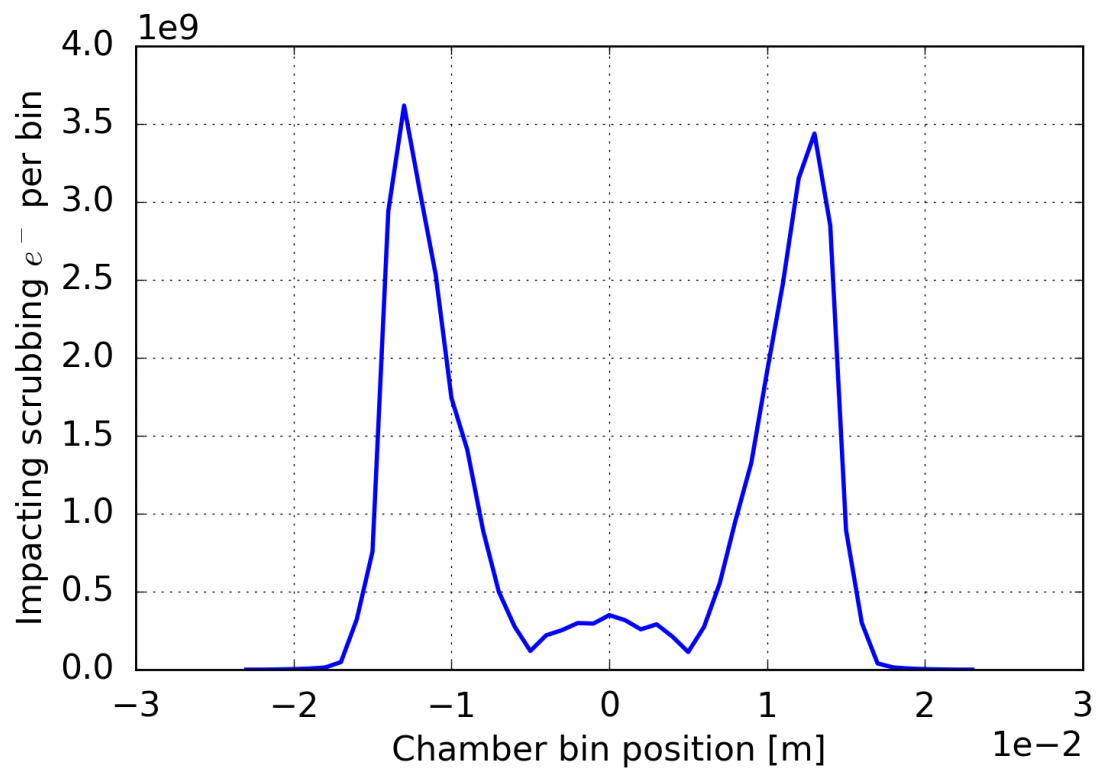
Var. name: sum(nel\_hist, axis=0)  
number of electrons in each slice

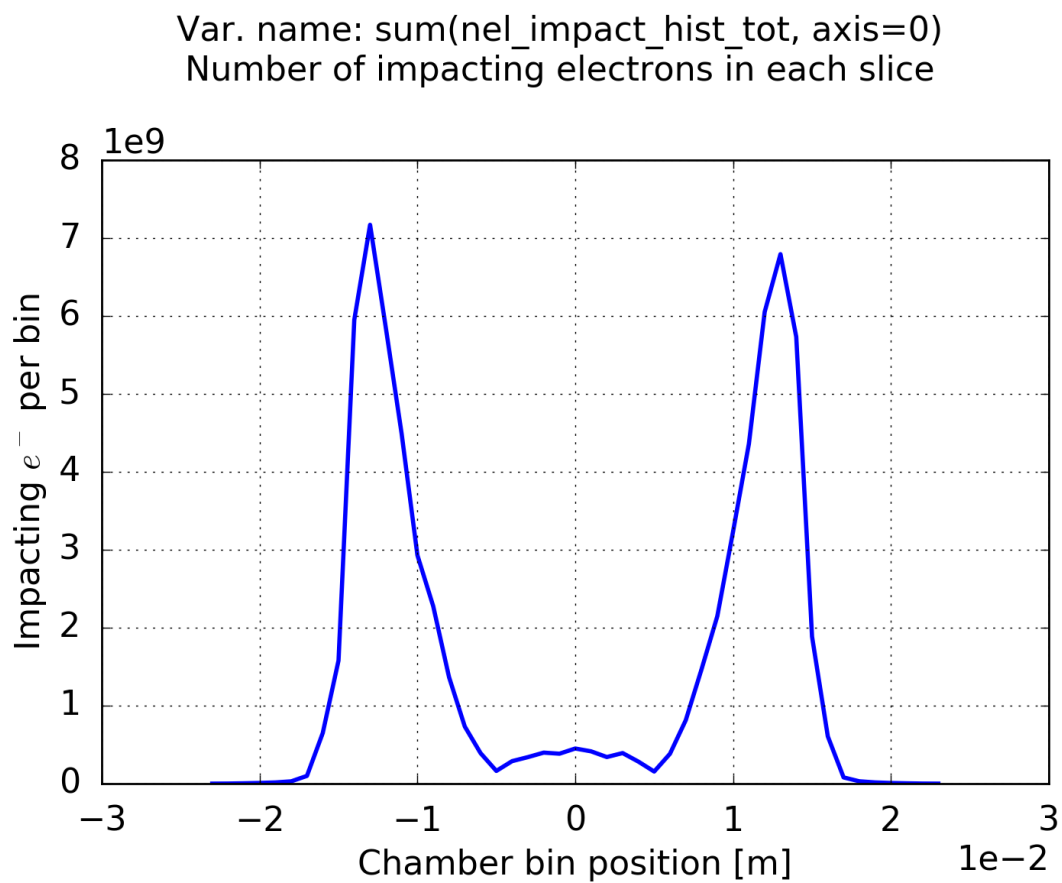
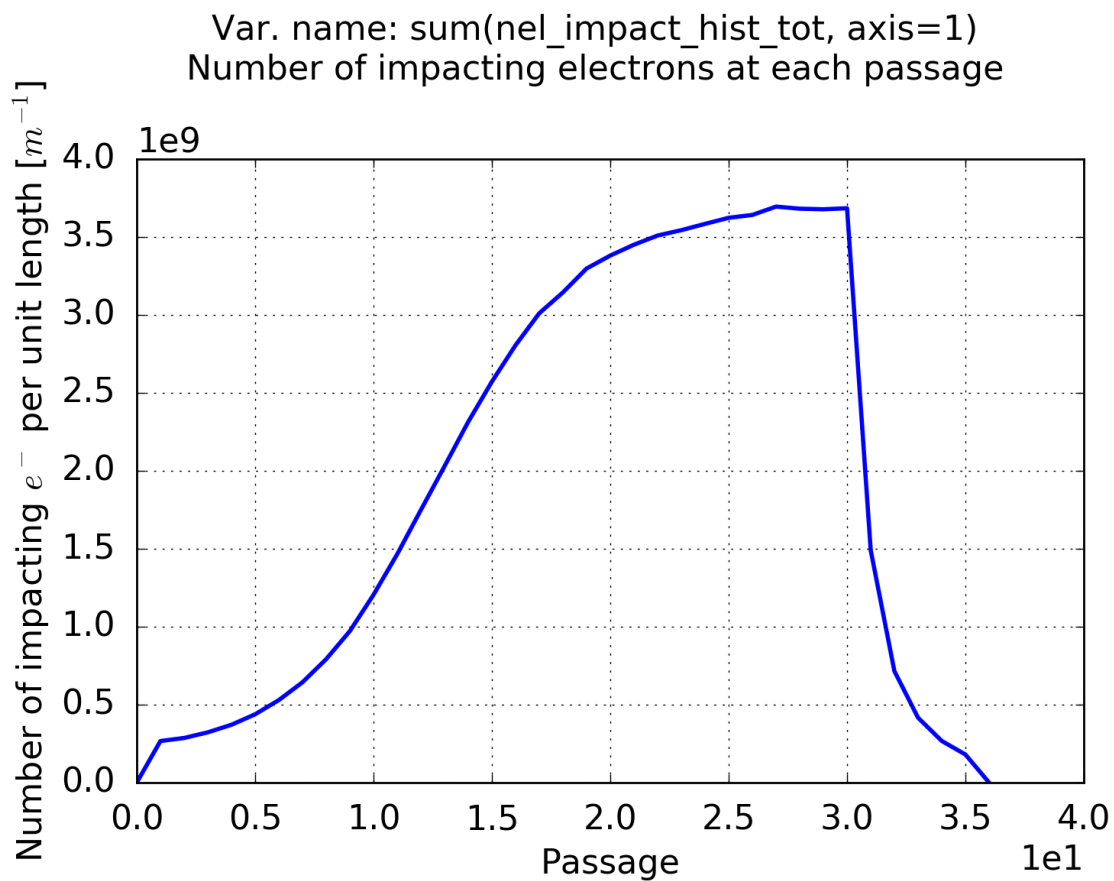


Var. name: np.sum(ob.nel\_impact\_hist\_scrub, axis=1)  
Number of impacting scrubbing electrons [ $E > E_{\text{scrub}}$ ]

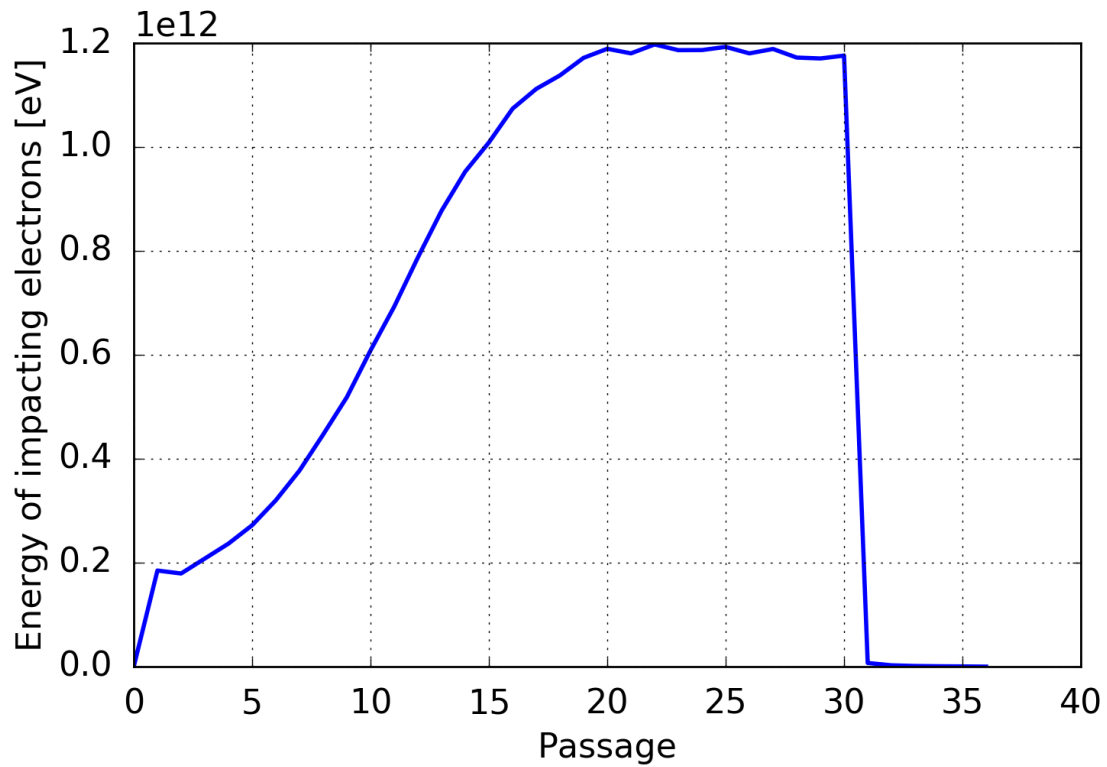


Var. name: sum(nel\_impact\_hist\_scrub, axis=0)  
Impacting scrubbing electrons in each slice





Var. name: sum(energ\_eV\_impact\_hist, axis=1)  
Energy of impacting electrons at each passage



Var. name: sum(energ\_eV\_impact\_hist, axis=0)  
Total energy of impacting electrons per passage [eV]

