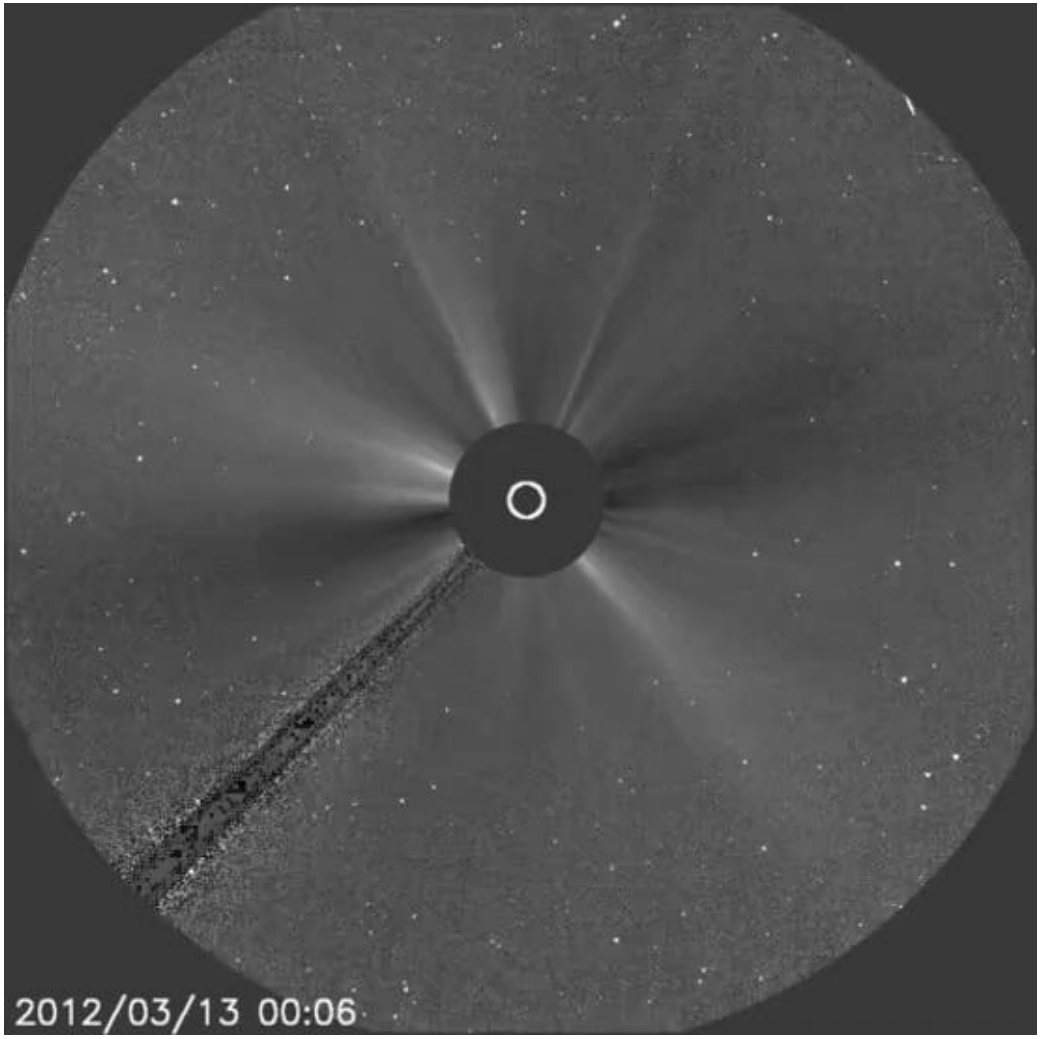
CCD Cosmic Ray Events Simulator Validation (using data from Gaia)

Cosmic ray events on astronomical CCDs

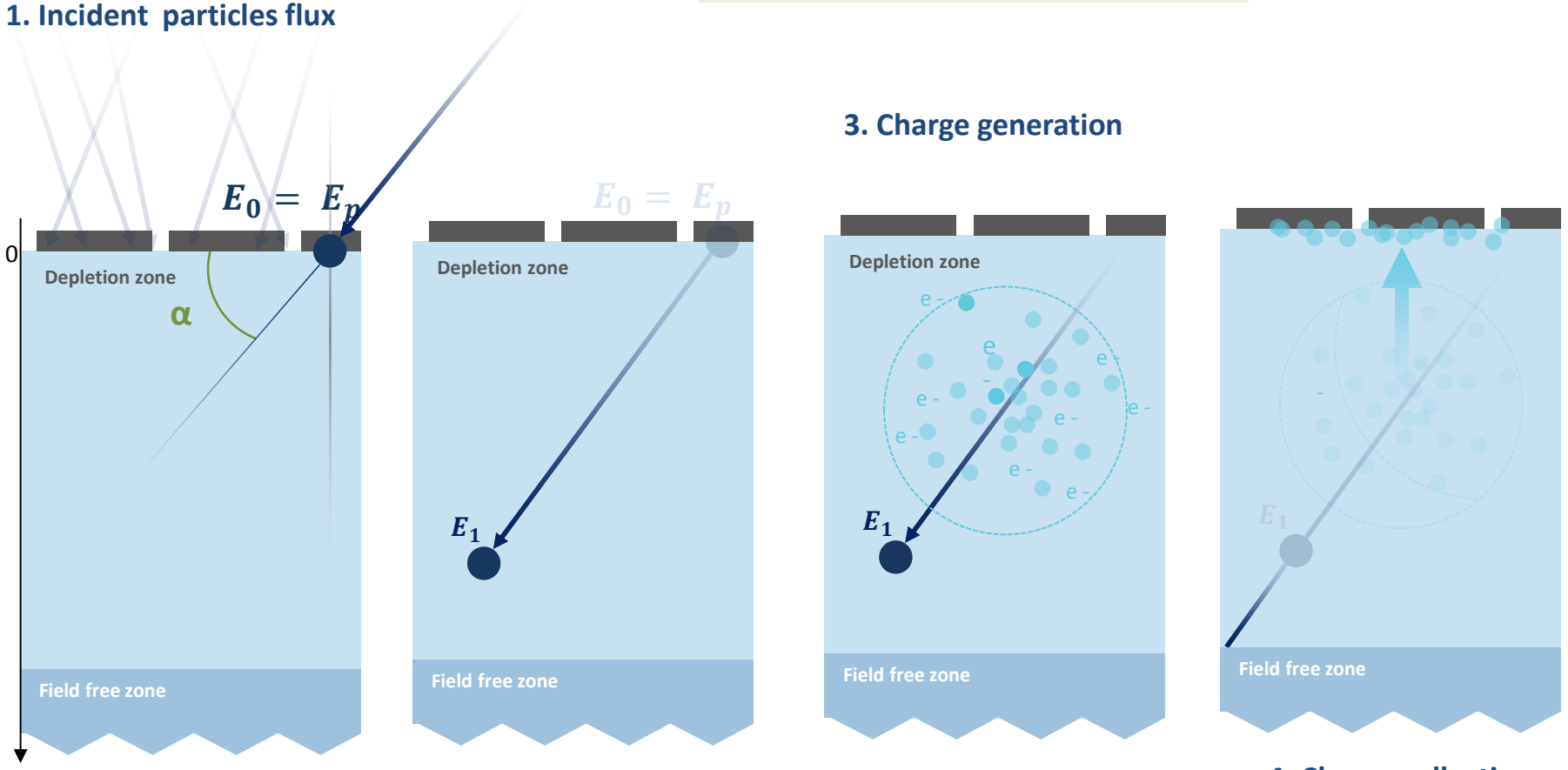


SOHO/LASCO C3 – 2012-02-13

Overview

For more details see :
[Physical model from the simulator]

1. Incident particles flux



2. Particle spreading

3. Charge generation

4. Charge collection

Physical model inside the simulator

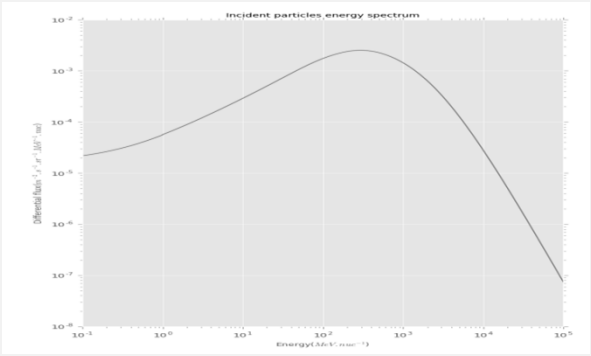
First implementation and hypothesis

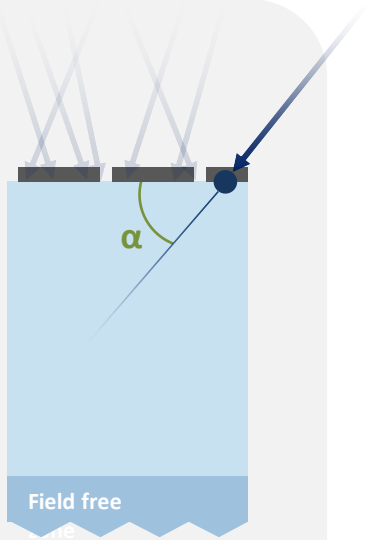
HYPOTHESIS

- Protons mainly responsible for prompt events (90% CRs)
- Isotropic incident angle distribution
- Protons are spreading into straight lines
- Stopping power = good approximation

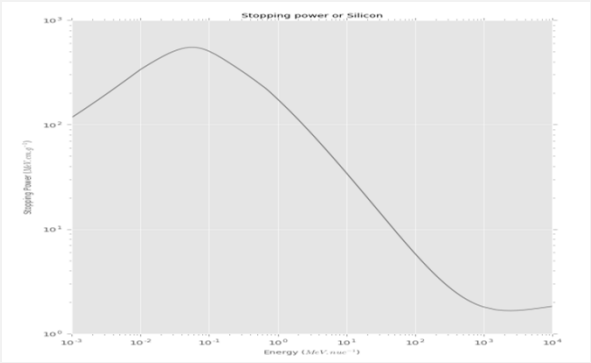
INPUTS

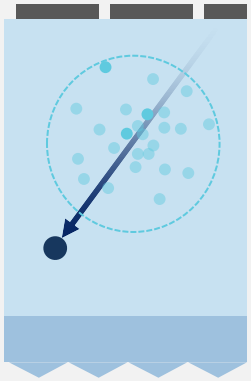
Flux from CREME96





Stopping power from NIST - EPSTAR

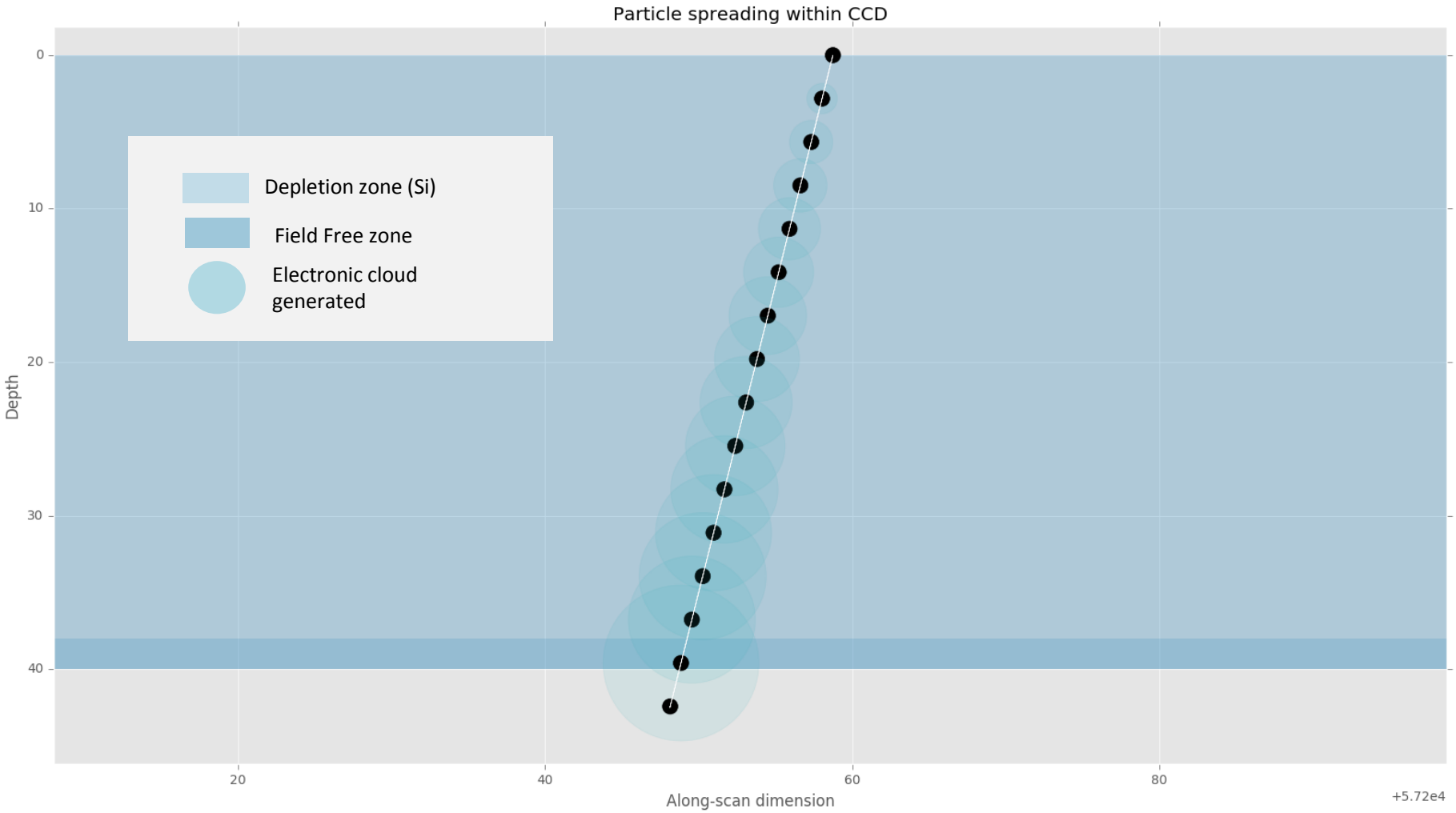




Stopping power curve → Particle energetic deposition in Si -> Charge generation

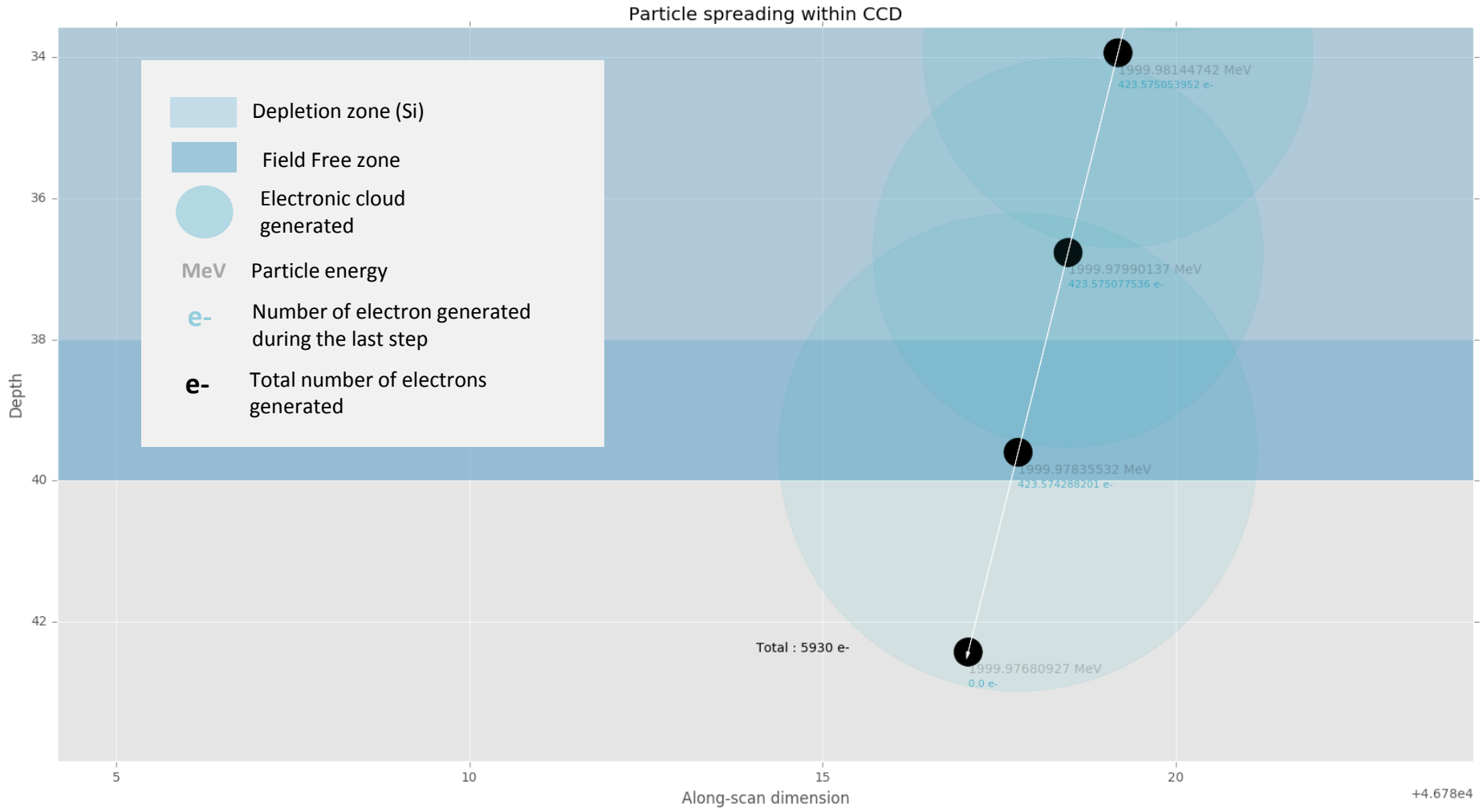
First implementation of T.A.R.S.
(Tools for Astronomical Radiation Simulations)

Example from the simulator with an incoming particle of E_p = 2000 MeV with $\alpha = {}^{\pi}/_{4}$ and a spreading step of $4\ \mu m$



First implementation of T.A.R.S.
(Tools for Astronomical Radiation Simulations)

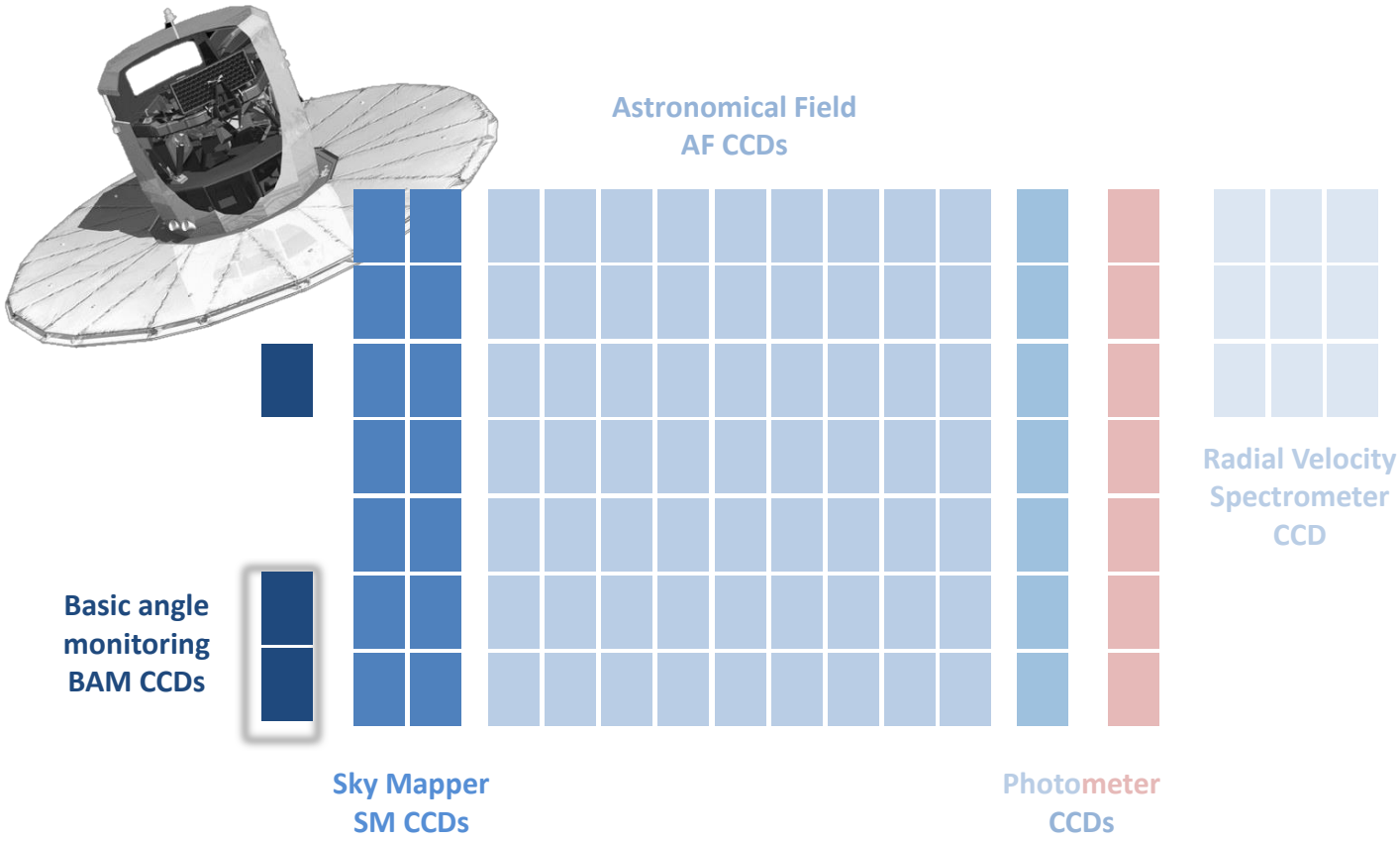
Example from the simulator with an incoming particle of E_p = 2000 MeV with $\alpha = \pi/4$ and a spreading step of $4\ \mu m$



How to validate this simulator?

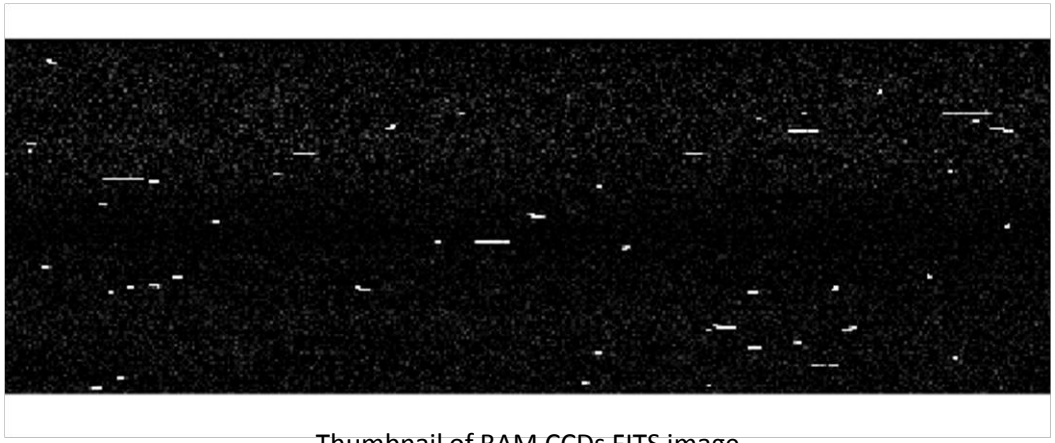
(Using Gaia Data…)

Gaia focal plane
What are we going to study?

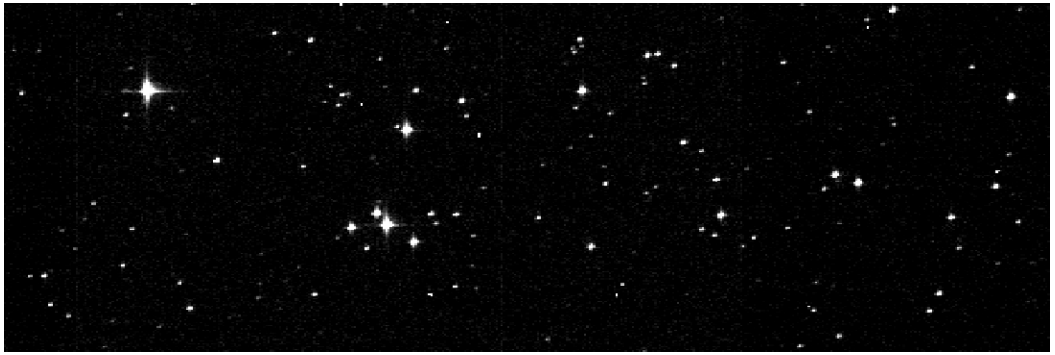


BAM CCDs vs. SM CCDs

What are we going to study?



Thumbnail of BAM CCDs FITS image



Thumbnail of SM CCDs FITS image

ADVANTAGES of BAM

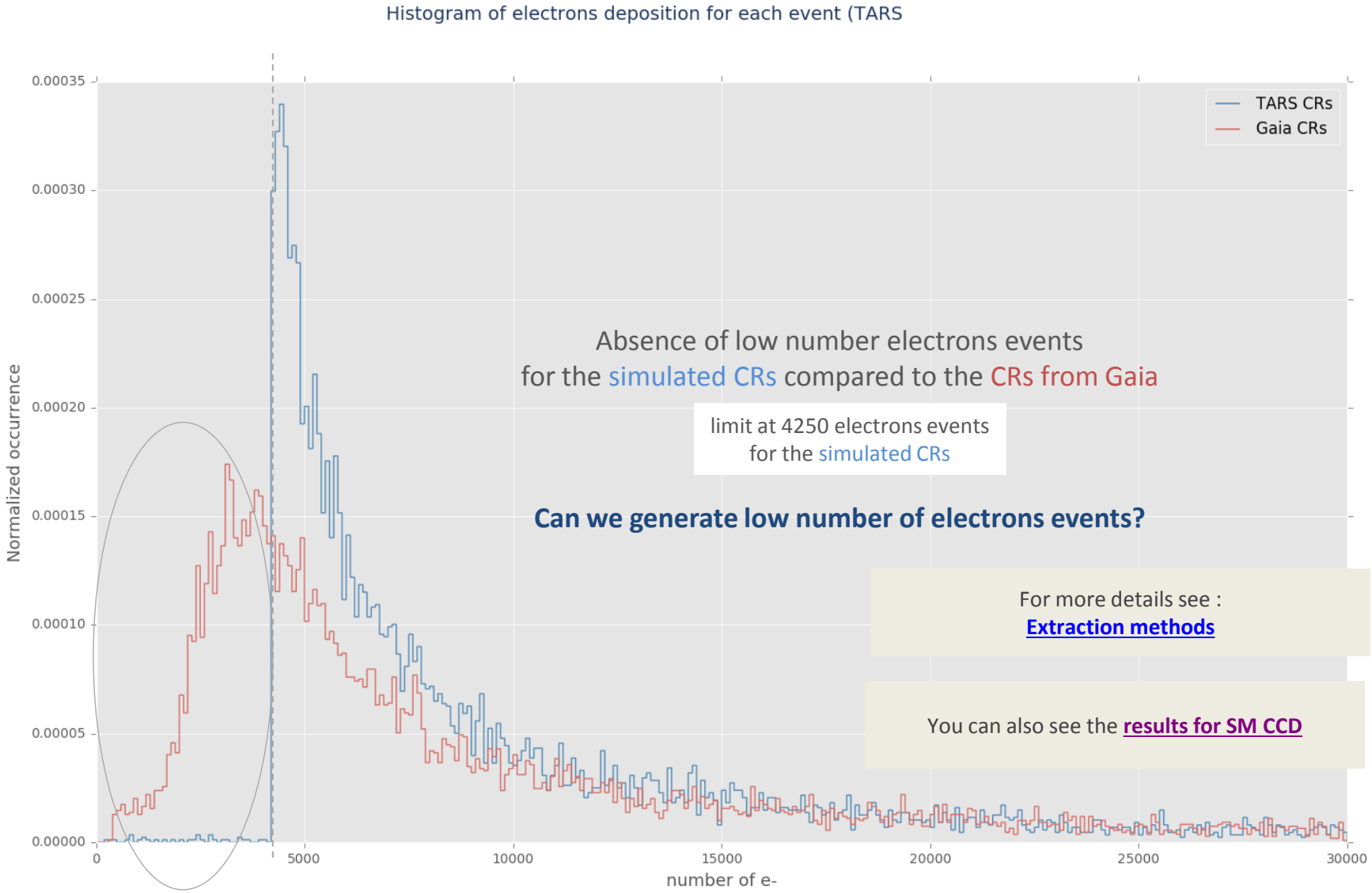
- More Cosmic rays events than SM (thickness : $40\ \mu m$)
- No stars

INCONVENIENTS of SM

- Less Cosmic rays events than BAM (thickness : $16\ \mu m$)
- Stars

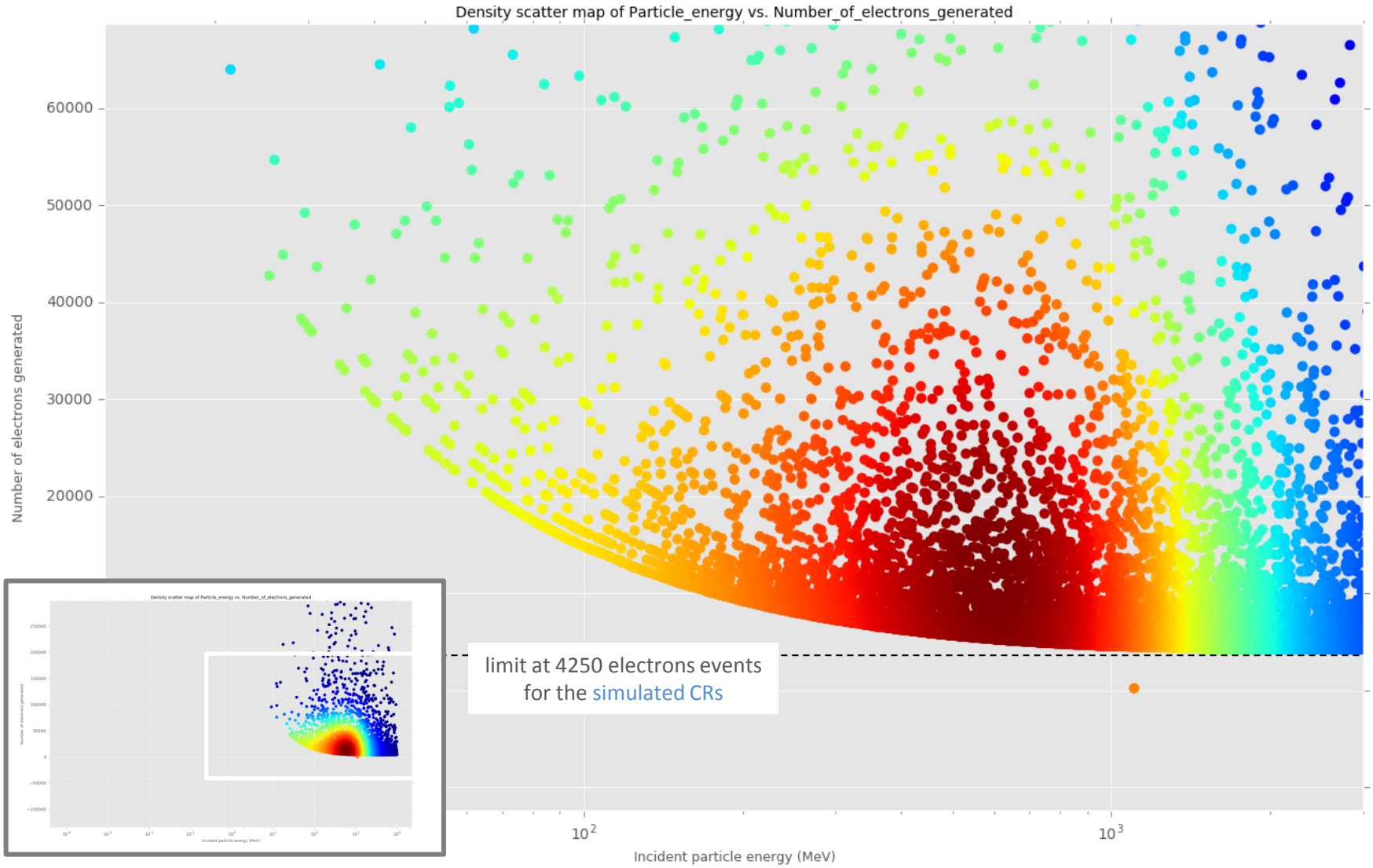
Firsts Results for BAM

Histogram



Firsts Results for BAM

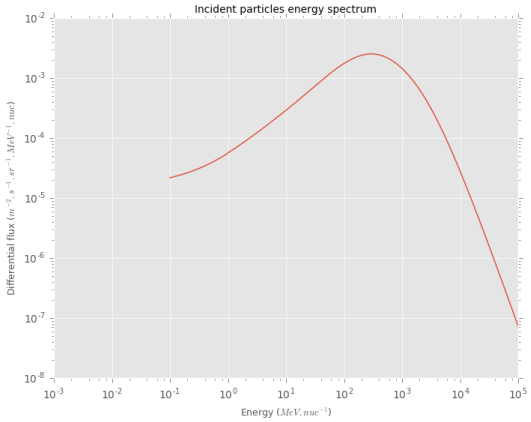
Scatter plot

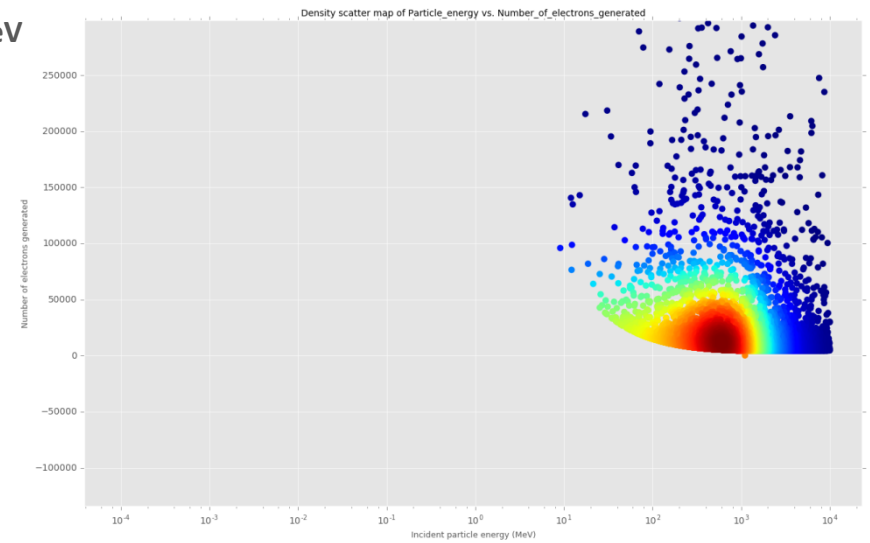


Firsts Analysis

CREME96 input Spectrum

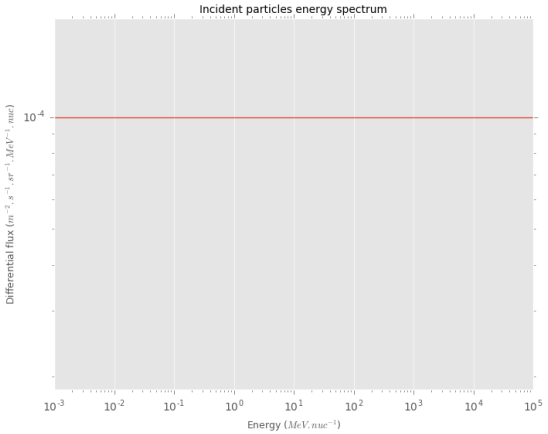
CREME96 particles energy spectrum from **100keV to 10 000MeV**

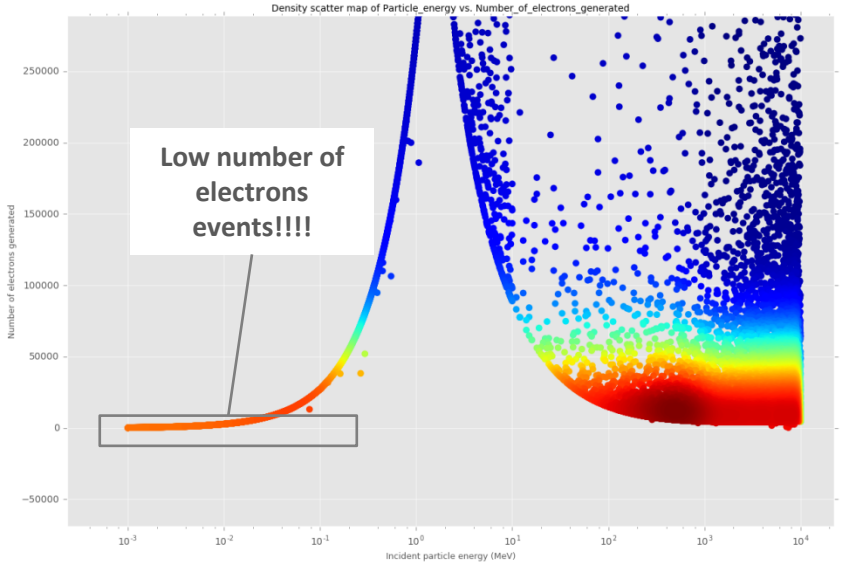




Uniform input spectrum for testing:

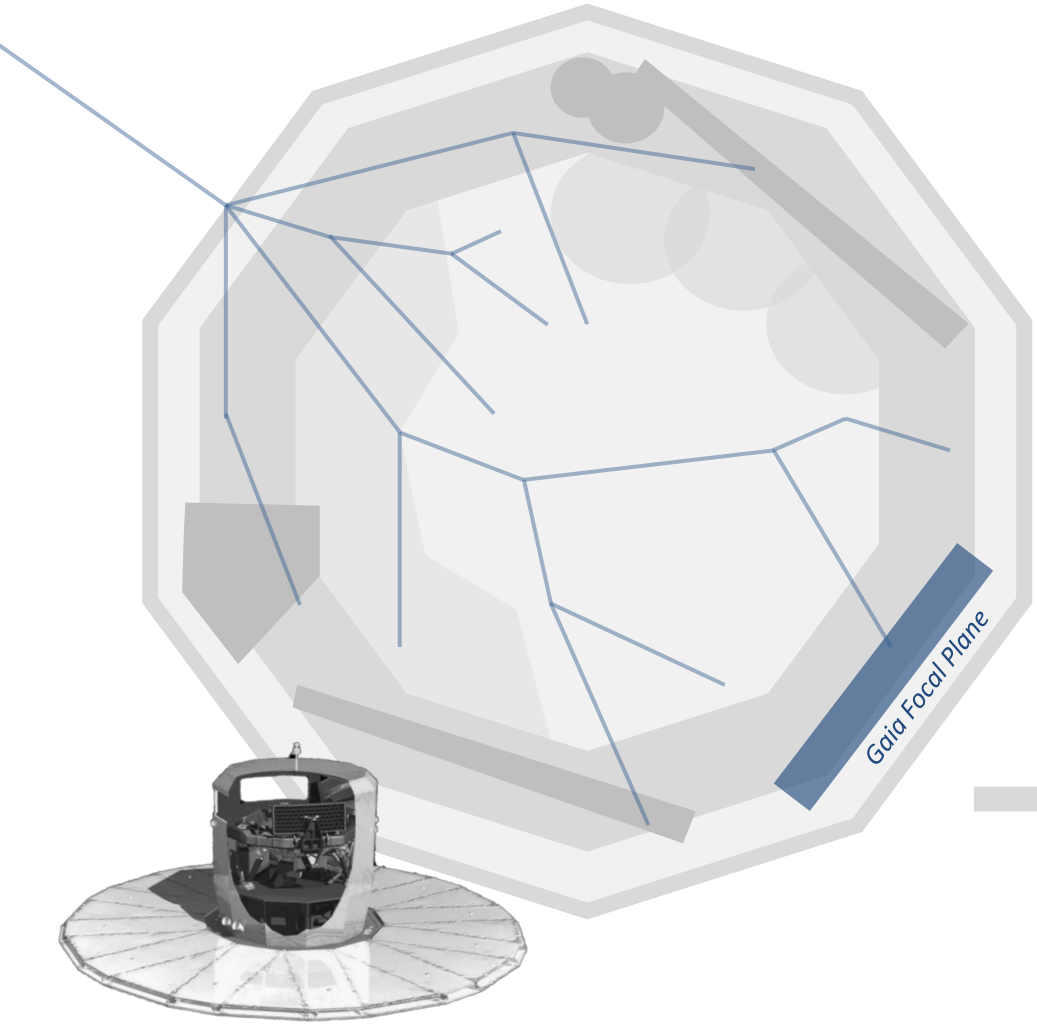
Wide particles energy spectrum from **1keV to 10 000MeV**





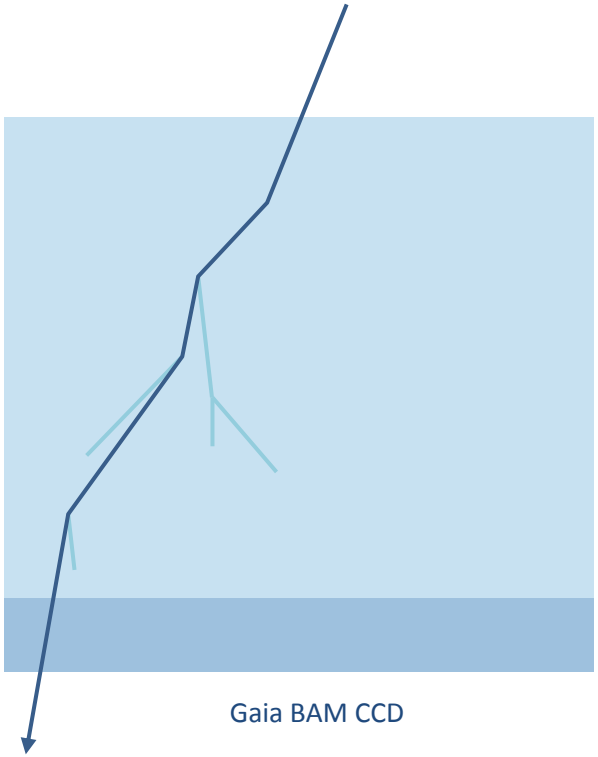
We need a more accurate and realistic input spectrum!!!!

Simulation of Gaia CRs impacts with GRAS and SPENVIS
(Geant4 Packages)



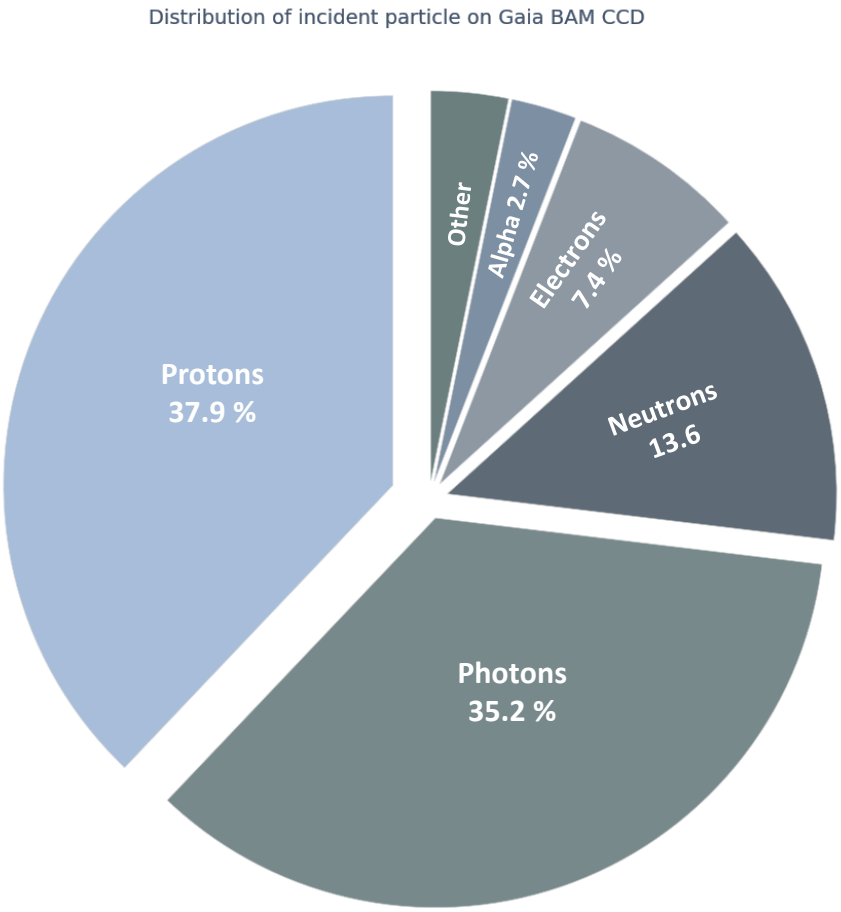
X 1 000 000 000

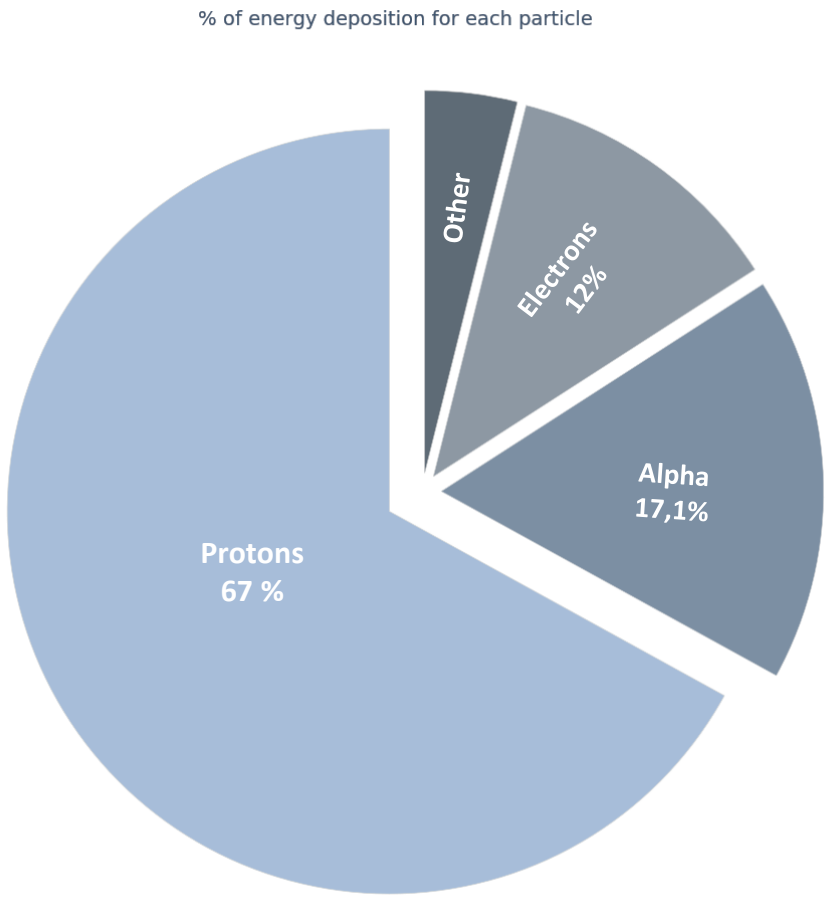
90% Protons (H^+)
10% Helions (He^{2+})



Simulation of Gaia CRs impacts with GRAS and SPENVIS

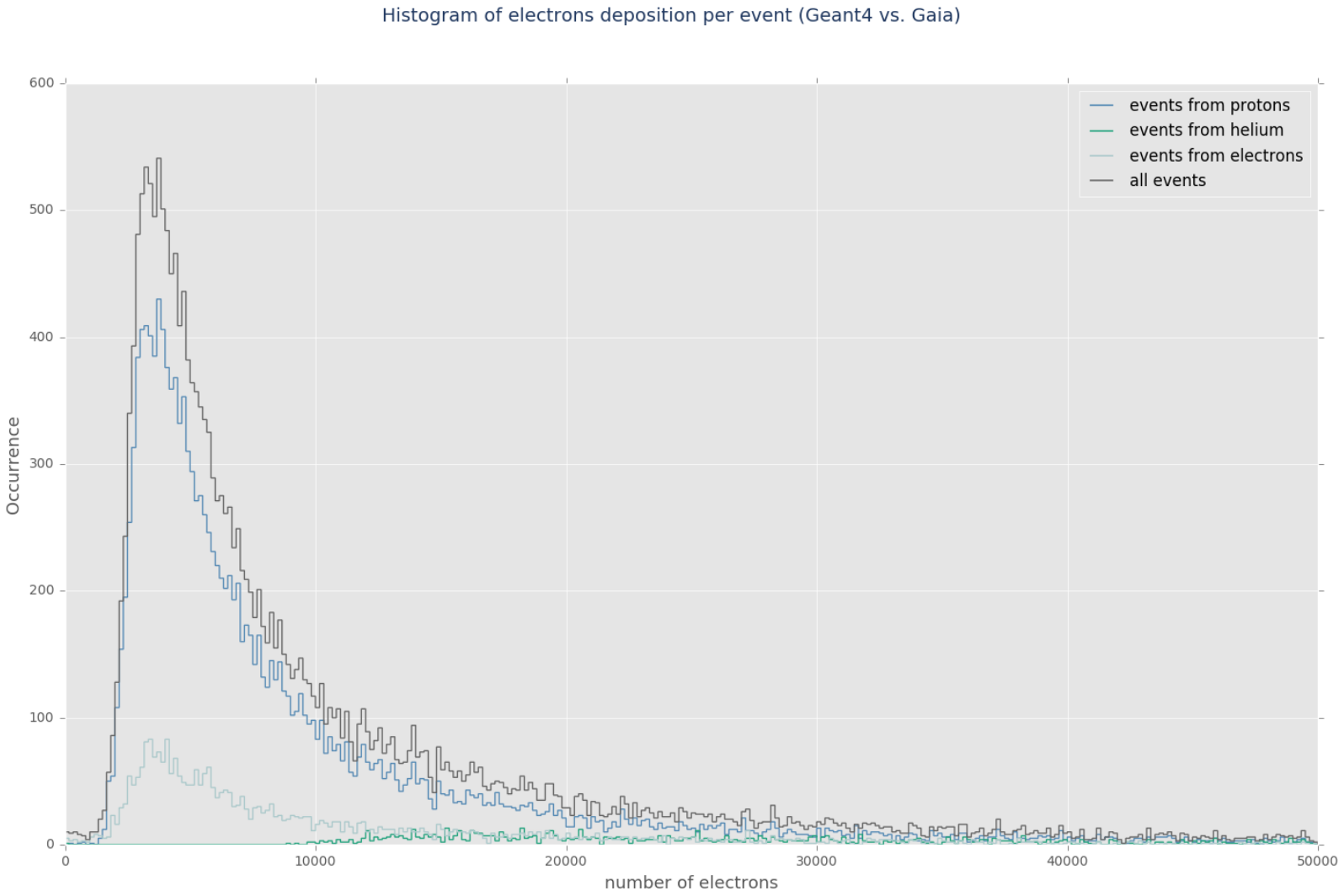
Particles proportions





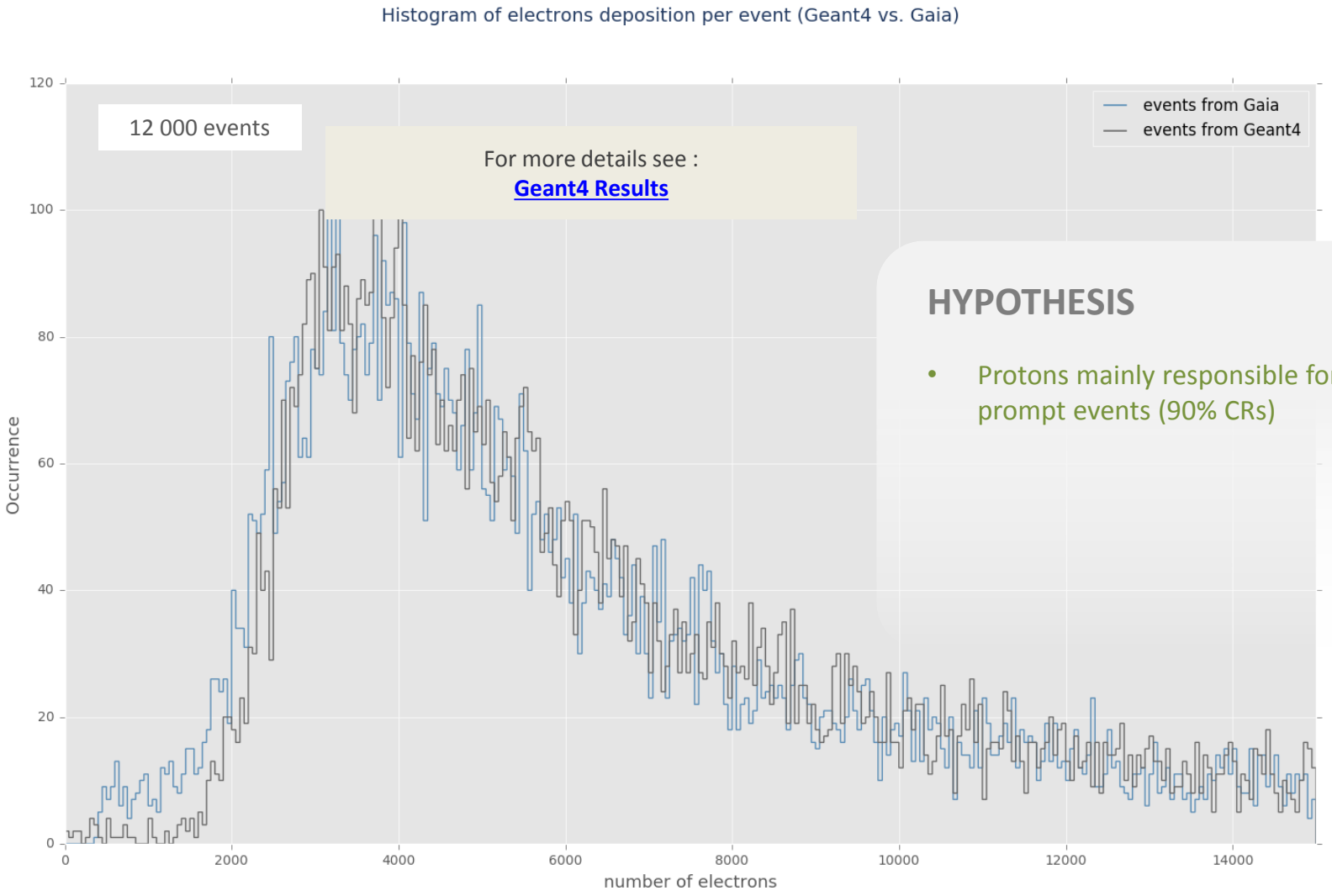
Simulation of Gaia CRs impacts with GRAS and SPENVIS

Particles proportions



Simulation of Gaia CRs impacts with Geant4

Results – Comparison with Gaia – Prompt events

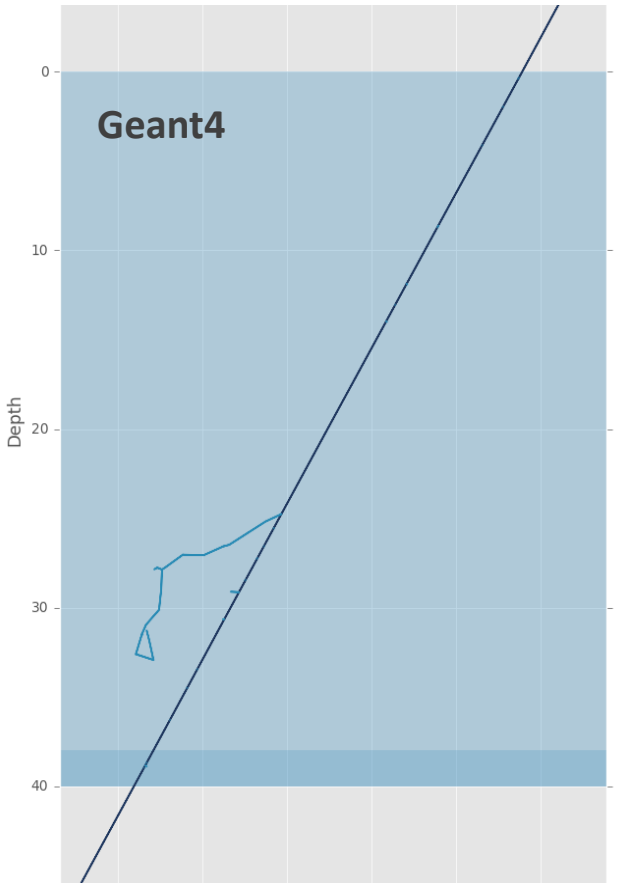


HYPOTHESIS

- Protons mainly responsible for prompt events (90% CRs)

Simulation of Gaia CRs impacts with Geant4

Results – Comparison with TARS – Protons spreading



HYPOTHESIS

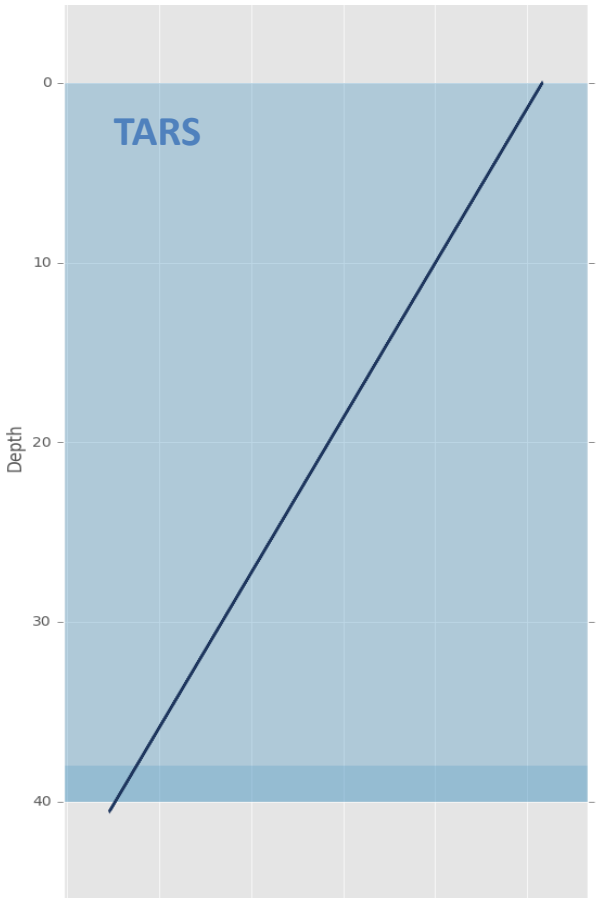
- Protons are spreading into straight lines

Depletion zone (Si)

Field Free zone

Proton trace in Si

Electron trace in Si



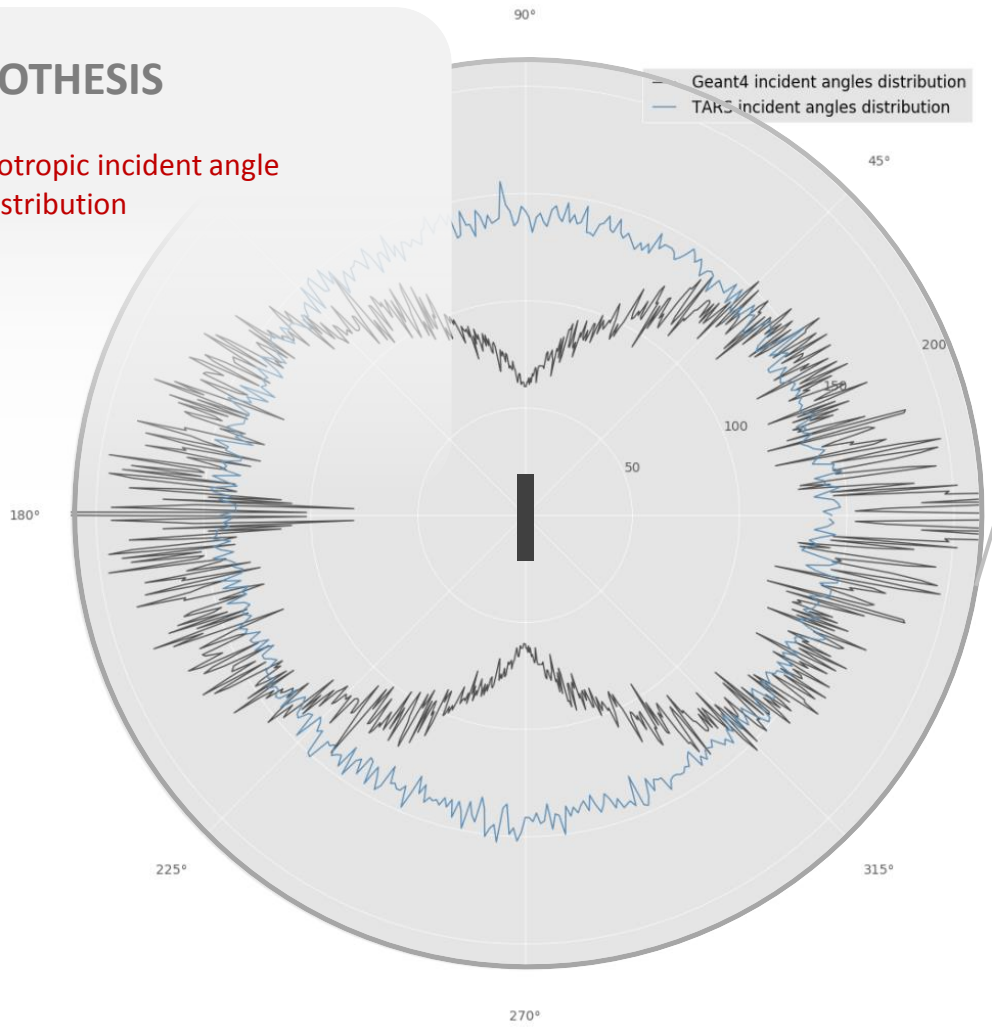
TARS simulation with same energies and incident angles as Geant4

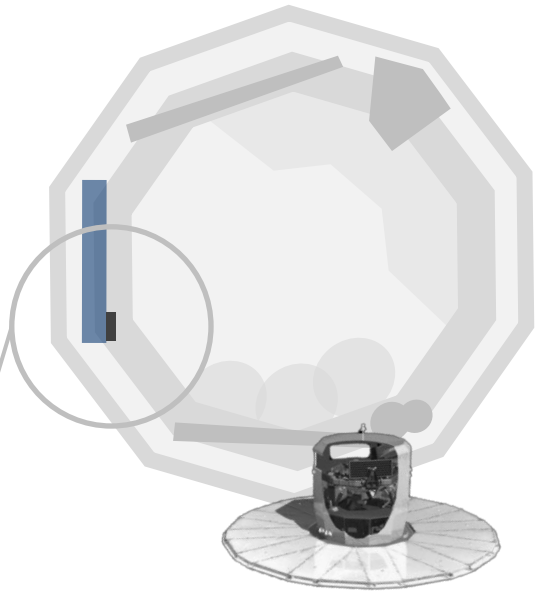
Simulation of Gaia CRs impacts with Geant4

Results – Comparison with TARS – Incident angles

HYPOTHESIS

- Isotropic incident angle distribution





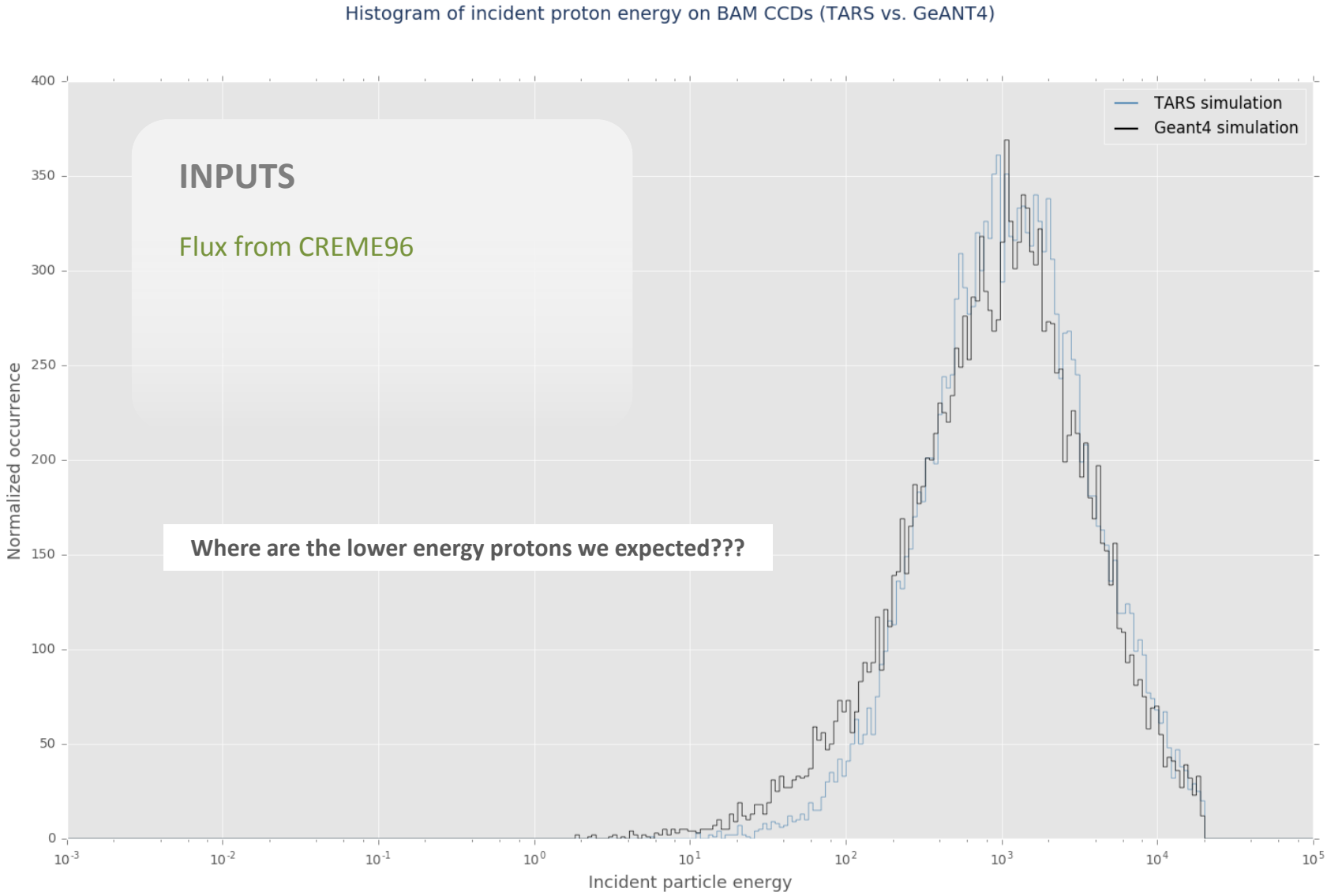
Gaia BAM CCD

Gaia Focal Plane

Incident angle distribution around Gaia BAM CCD

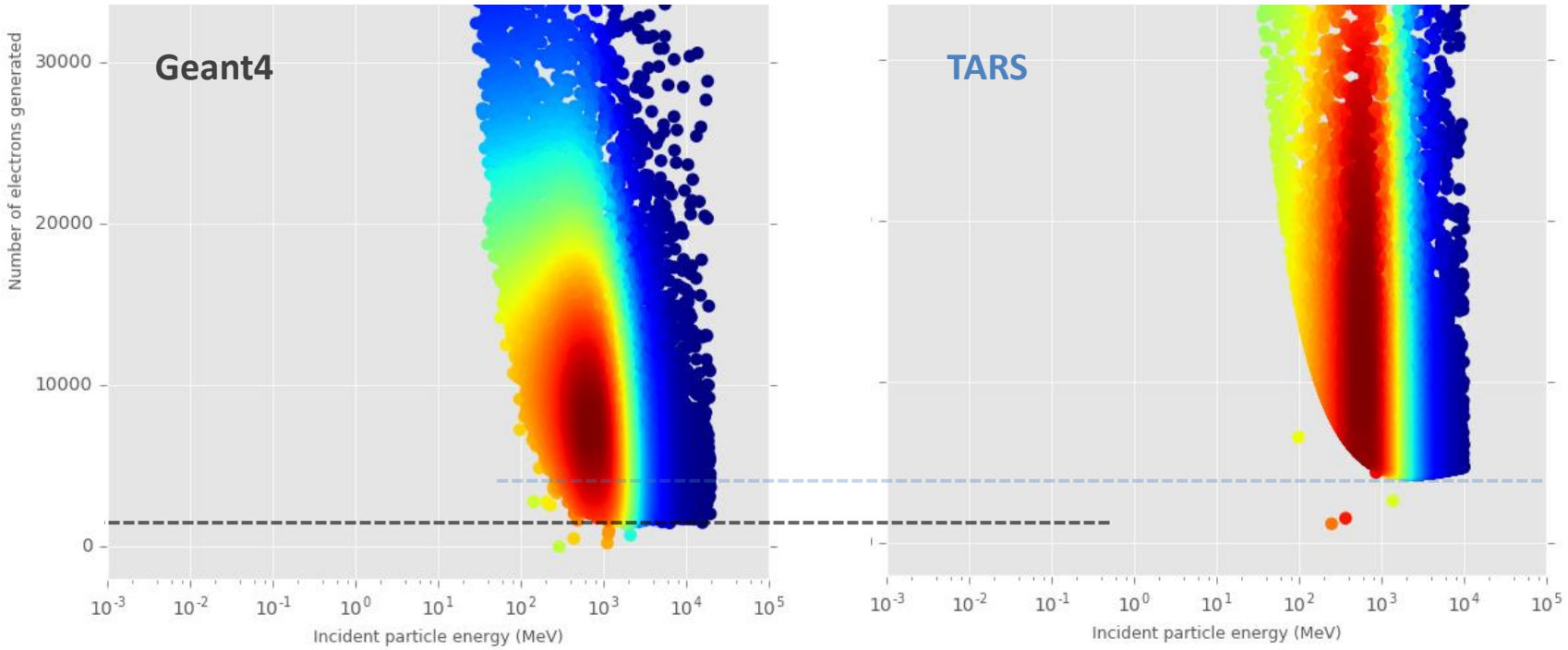
Simulation of Gaia CRs impacts with Geant4

Results



Simulation of Gaia CRs impacts with Geant4

Results – Comparison with TARS – Incident energy



limit at **1800 electrons** events for **Geant4 simulation**

limit at **4250 electrons** events for **TARS simulation**

Energetic deposition of protons is not the same

Simulation of Gaia CRs impacts with Geant4

Other Studies conducted during the internship

- CRs extraction validation
  L.A. cosmic library validation for BAM and SM

  For more details see :
  Extraction methods

- Model Parameters influence
  Spreading step, noise influence…etc.

  For more details see :
  Parameters influence

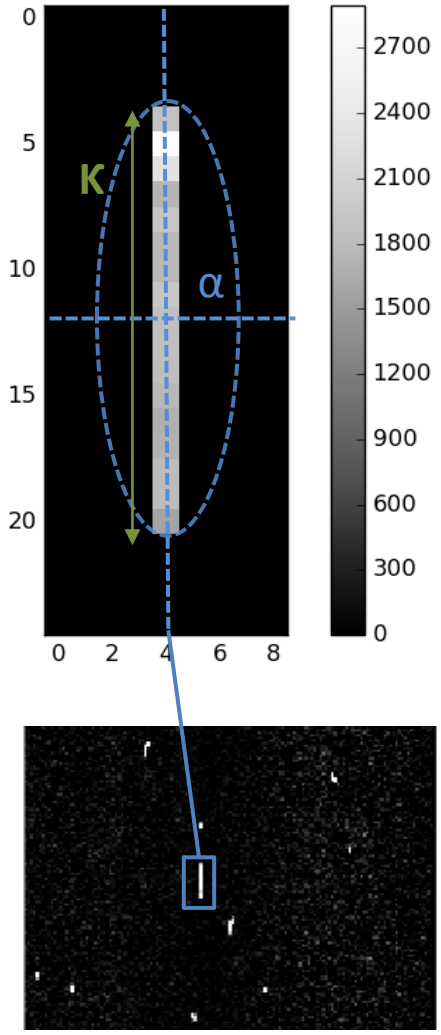
- FORTRAN vs. PYTHON performance
  Processing time, results accuracy, Translation

  For more details see :
  Fortran to Python

- Geometrical study of CRs events
  Orientation, length…etc.

- GCR protons, solar protons and
  Interplanetary electrons effects (Geant4)

  For more details see :
  Root analysis

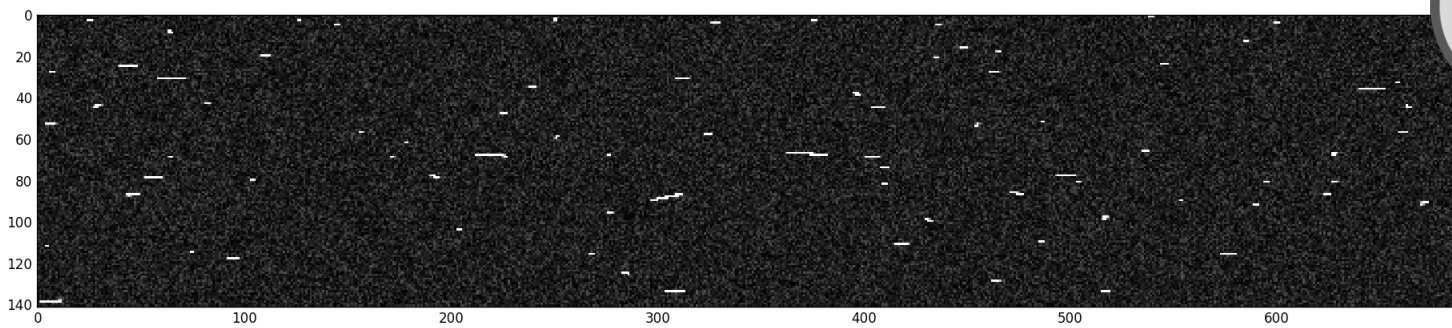


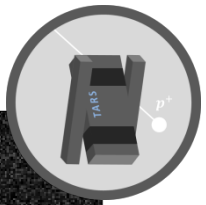
Example of geometrical characteristics studied

Simulation of Gaia CRs impacts with TARS + Geant4

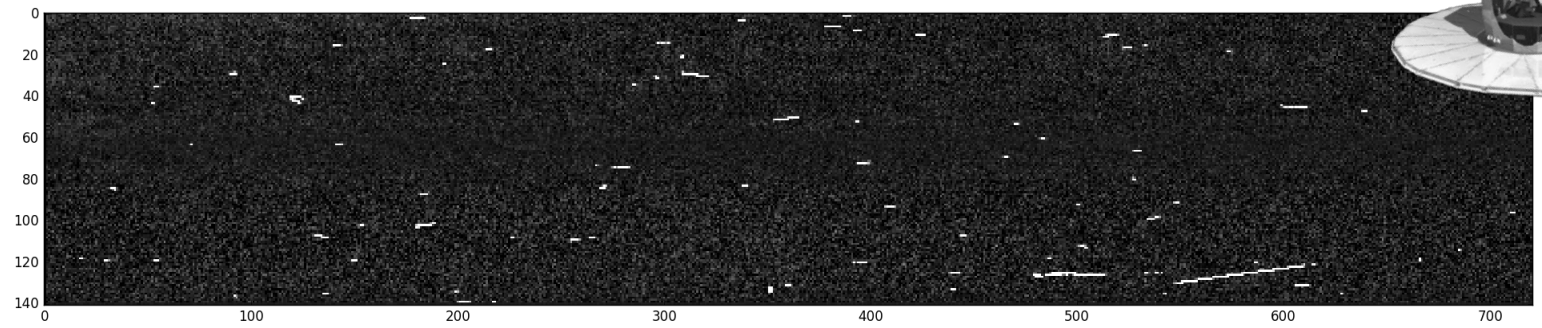
FITS simulation

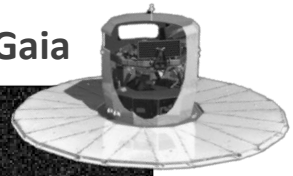
TARS (+ Geant4, GRAS, SPENVIS)





Gaia





Conclusion

What we thought

- Protons are mainly responsible for prompt events (90% CRs)
- Isotropic distribution of incident particle
- GCR protons are spreading into straight lines
- Stopping power = good approximation
- CREME96 inputs is adapted for our simulation
- Geant4 will be necessary

What we've learnt

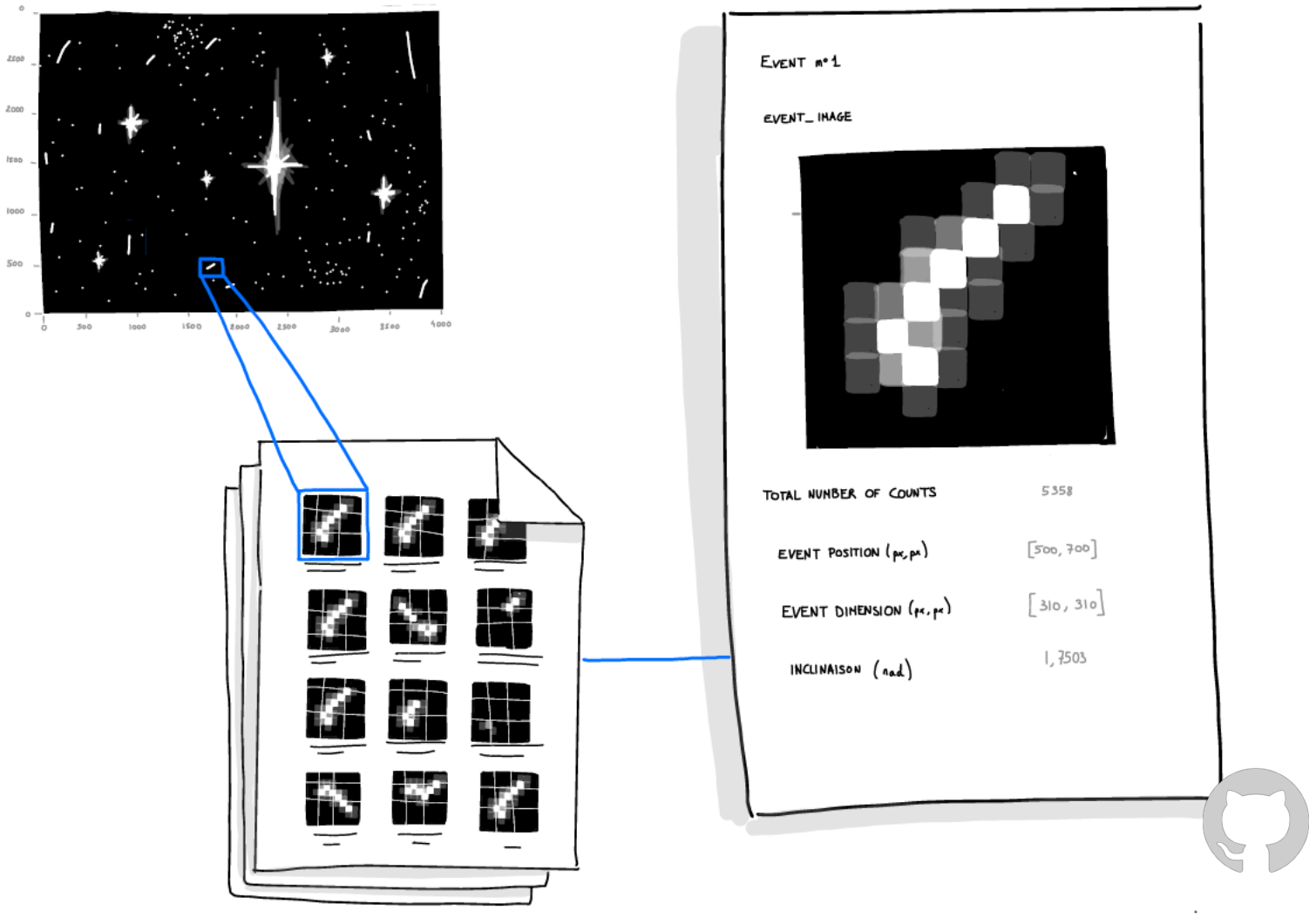
We confirm that Geant4 is needed to properly simulate CRs deposition

We also know that GRAS (Geant4 package) needs a 3D model of the spacecraft, not available at an early development phase, when some tests should be conduct. CREME96 is still right enough to do a simple expertise.

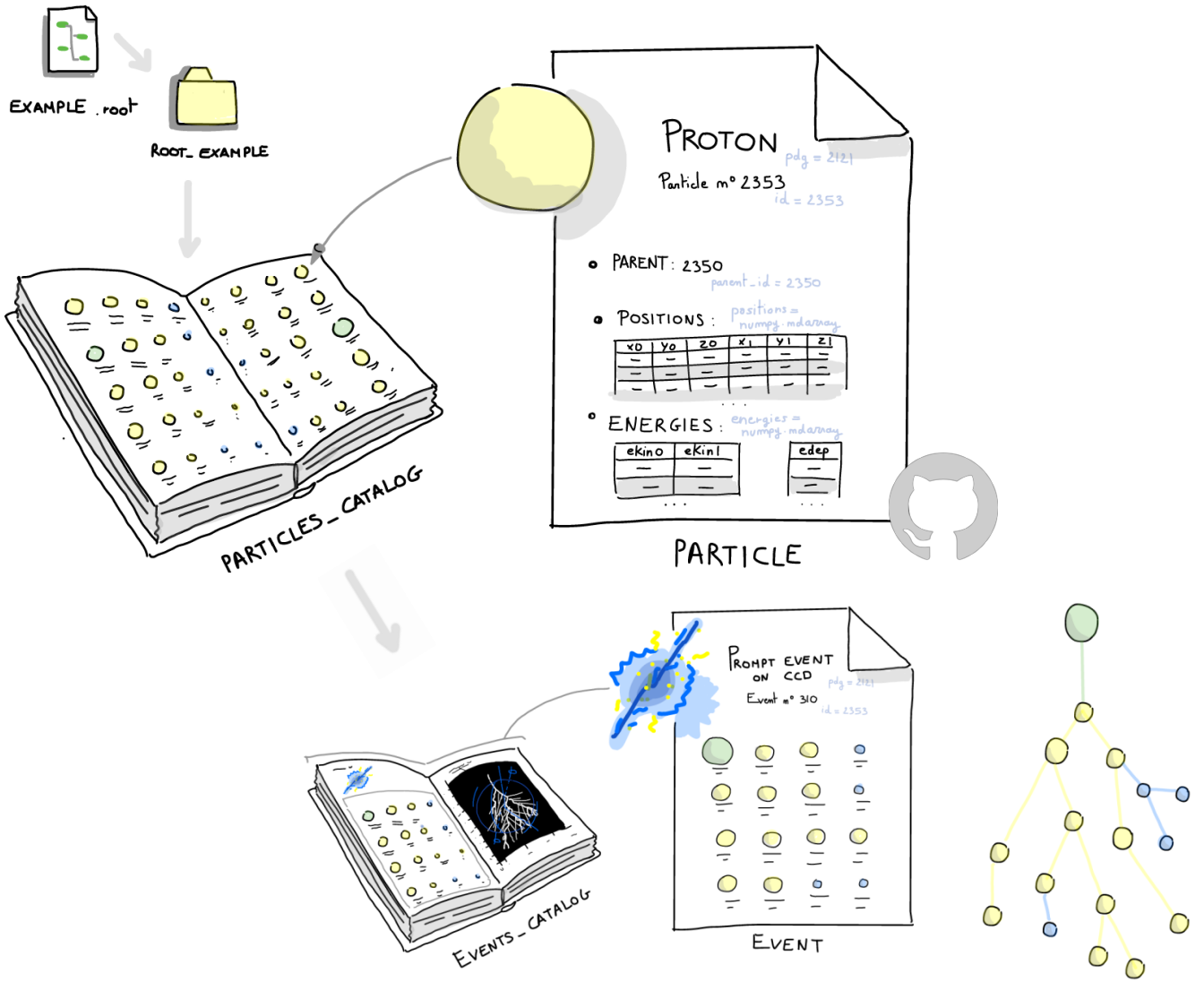
Python libraries development

Available on Github!

Cosmic rays extraction and study



ROOT file python analyser



Physical model from the simulator

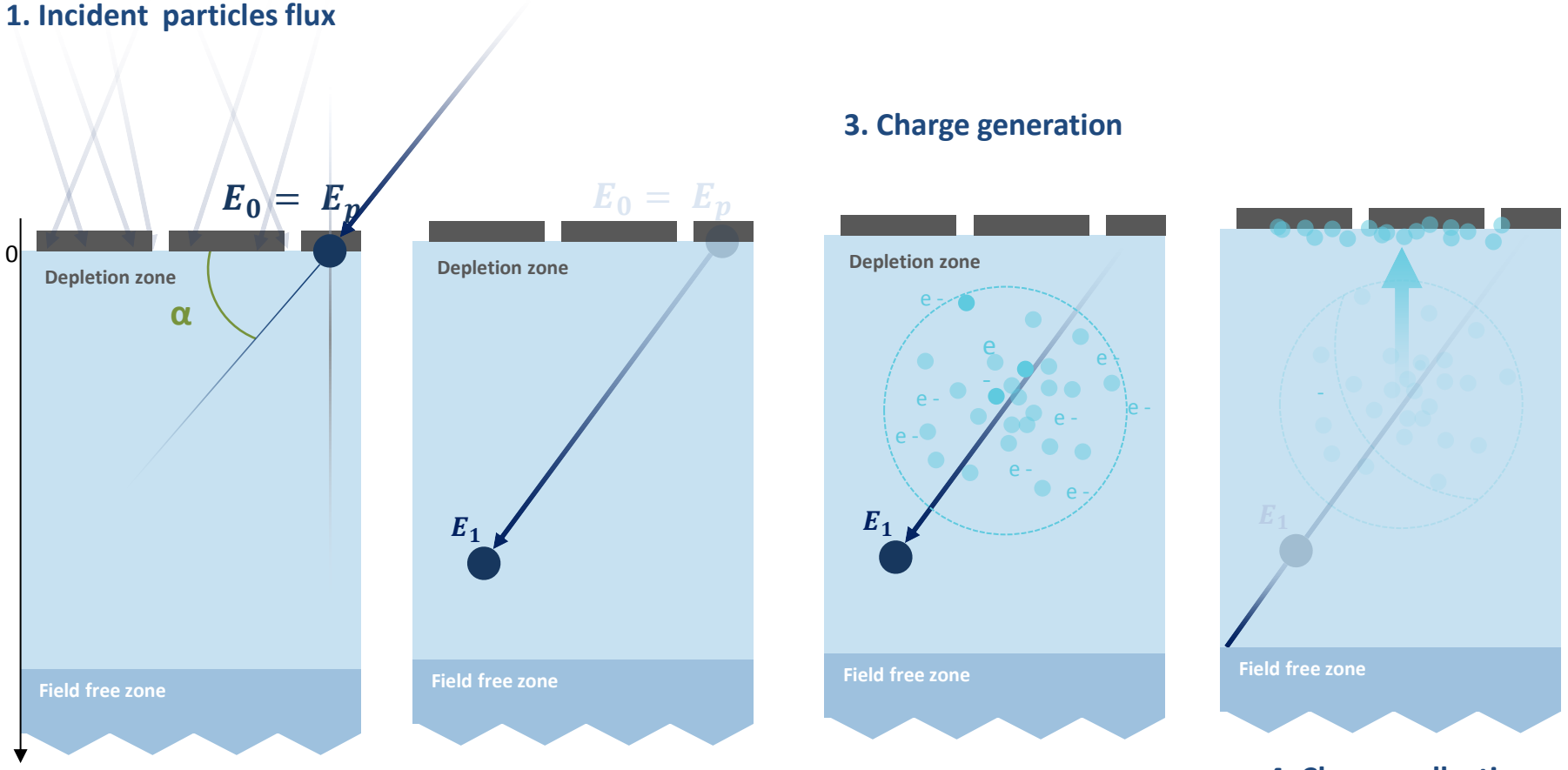
Abstract

The CR simulator we are using is based on a Fortran code written by Alex Short. To use this simulator correctly we have extracted the physical model from the code. This section is a description of this model base on papers and books from bibliography

Summary

Overview

1. Incident particles flux



2. Particle spreading

1. Constants definition

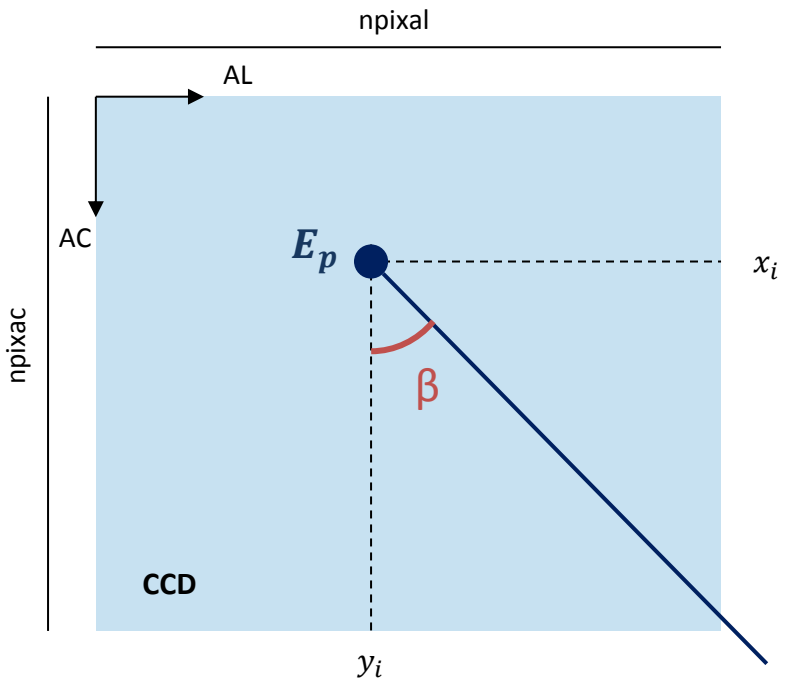
To begin with the physic behind the simulator, we have first to some physical constants :

k_B	$1{,}38.10^{-23}\ m^2.kg.s^{-2}.K^{-1}$	Boltzmann constant
$q(e-)$	$1{,}16.10^{-19}\ C$	Electron charge
ε_0	$8.85.10^{-12}\ F.m^{-1}$	Relative permittivity of Si
$\varepsilon_{Si}/\varepsilon_0$	11.8	Relative permittivity of Si
E_{e-h}	$3.65\ eV.electrons^{-1}$	energy required to generate an electron-hole pair
n_{Si}	$2.3290\ g.cm^{-3}$	Silicon density

And in our context, we also define :

N_a	1.10^{19}	The atomic density
l_1	1000	The diffusion length (field-free region)
b	2	depletion/field free boundary parameter

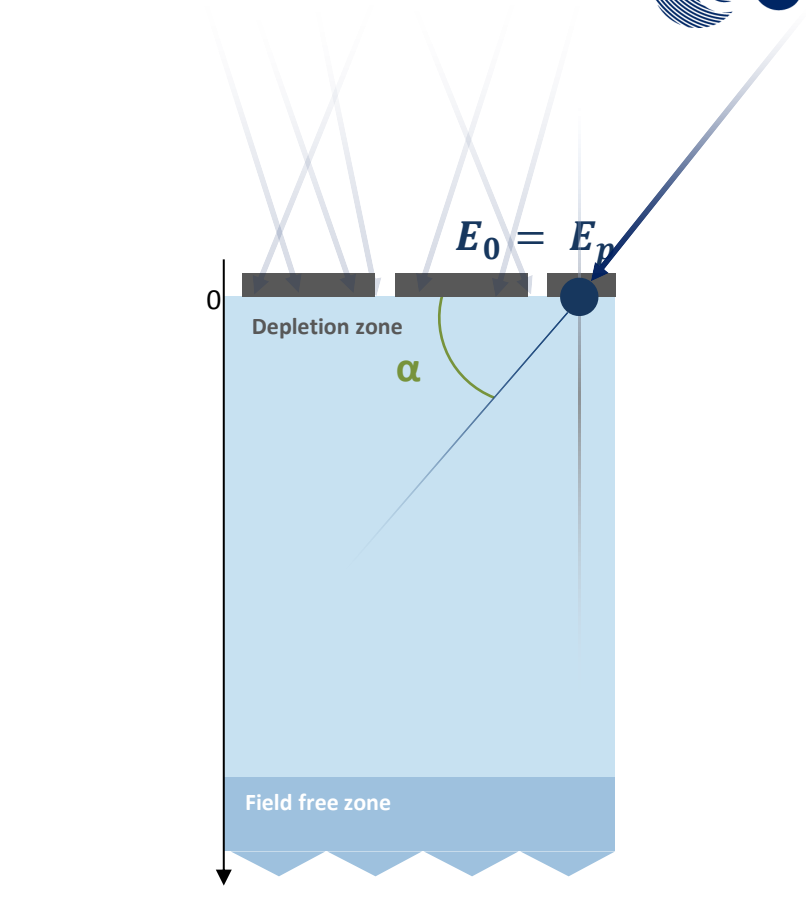
2. Random particle generation



Generation of a particle with random parameters :

- Energy (eV) : E_p
- Angles (rad) : β et α
- Position () : $(x_i,\ y_i)$

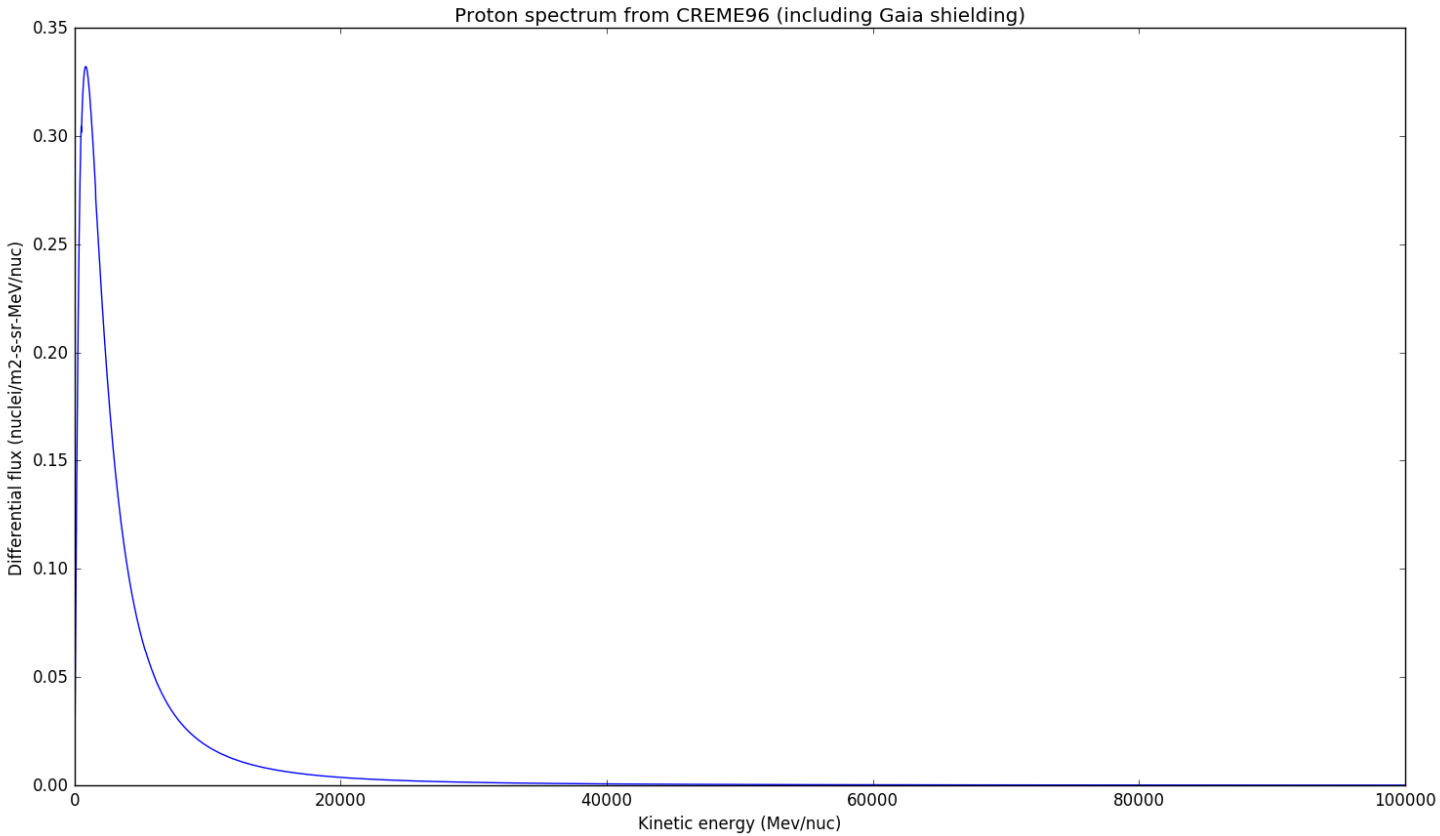
Considering L2 environment, 90% are protons and 10% are Helium



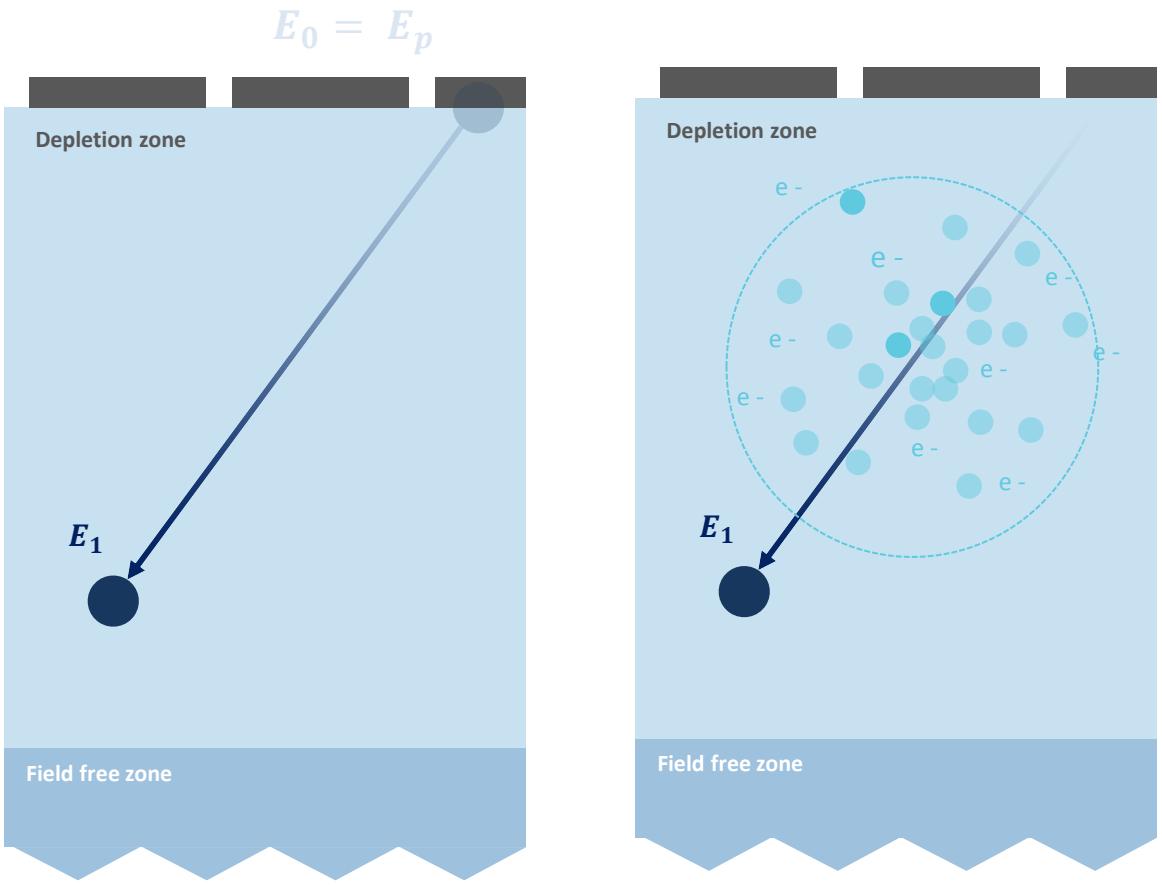
We represent here a front-side illuminated CCD but we can extend the model to back-side illuminated CCD (as Gaia's one) We took Gaia's data to estimate the thickness of each zone (substrate thickness is really exaggerated)

2. Random particle generation

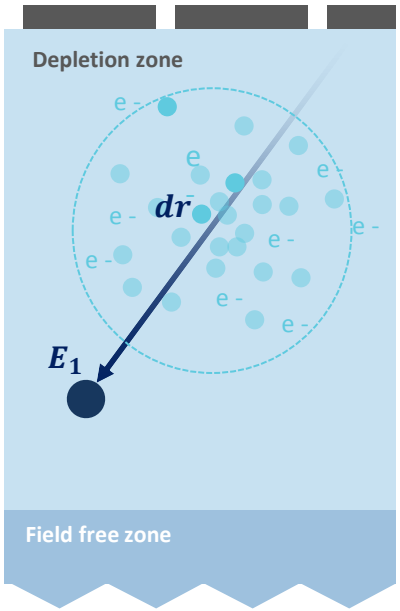
Random energy E_p is generated from CREME96 particle distribution for proton an helium (including the Gaia shielding of 11mm)



3. Particle spreading



4. Charge generation



We take as an example the first step of the particle spreading into the depletion zone.

We have :

$$dr = (dx, dy, dz) \ \ with \begin{cases} dx = \cos\alpha \cos\beta \\ dy = \cos\alpha \sin\beta \\ dz = \sin\alpha \end{cases}$$

During it displacement, the particle looses ΔE and generate a cloud of $N(e-)$ electrons with (From Scientific CCD, Janesick) :

$$N(e-) = \frac{\Delta E}{E_{e-h}}$$

One of the input of our model is the massive stopping power $S_m(E) = -dE/dr$ (here in $MeV.cm^2.g^{-1}$)

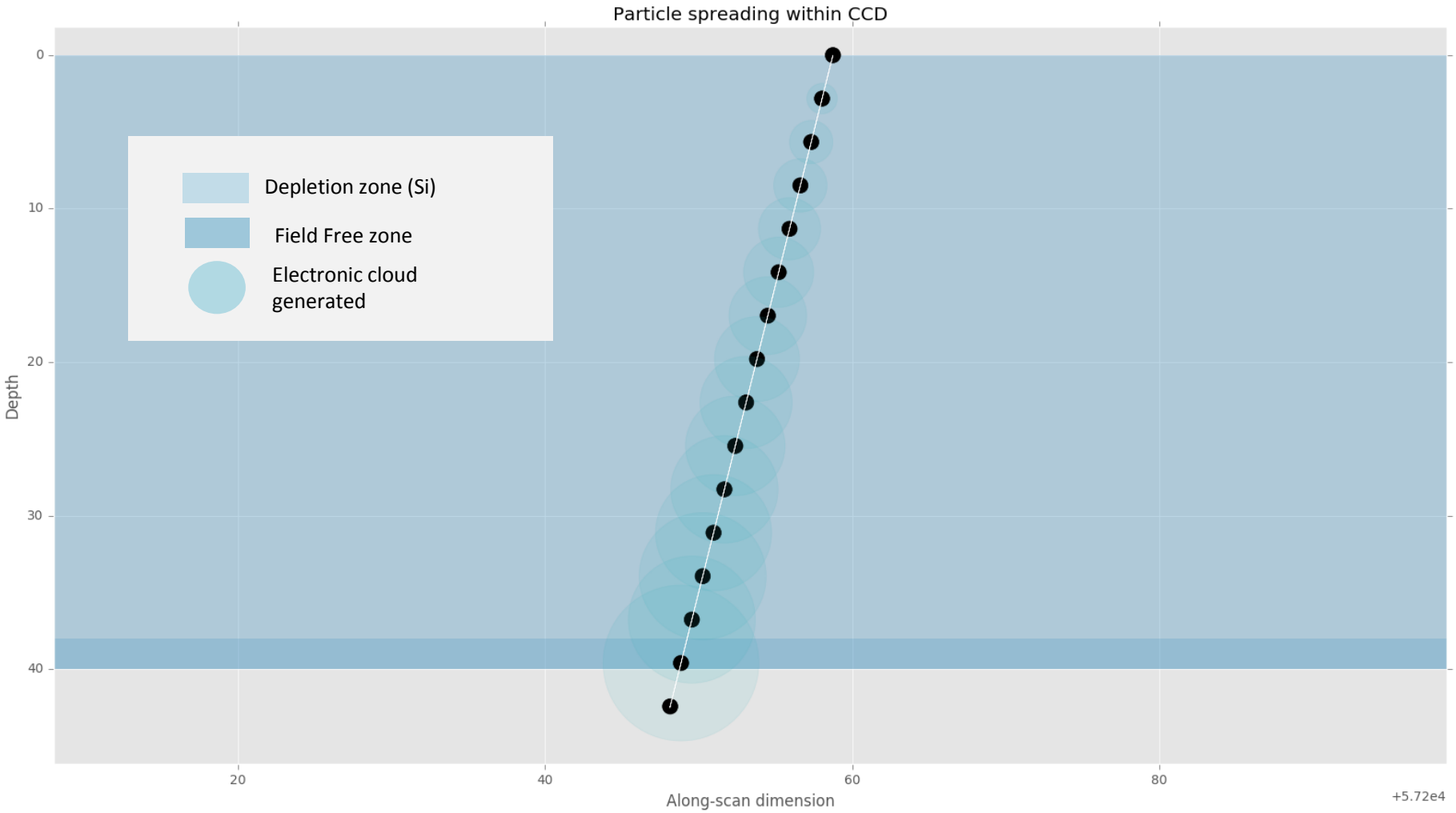
So we have :

$$\Delta E = \boxed{10^2 \, S_m(E_0) \ \ n_{Si}} * \boxed{1 \ \mu m}$$

$S_m(E_0)$ *in* $MeV.cm^{-1}$ so $10^2 \, S_m(E_0)$ *in* $eV.\mu m^{-1}$ — Spreading step

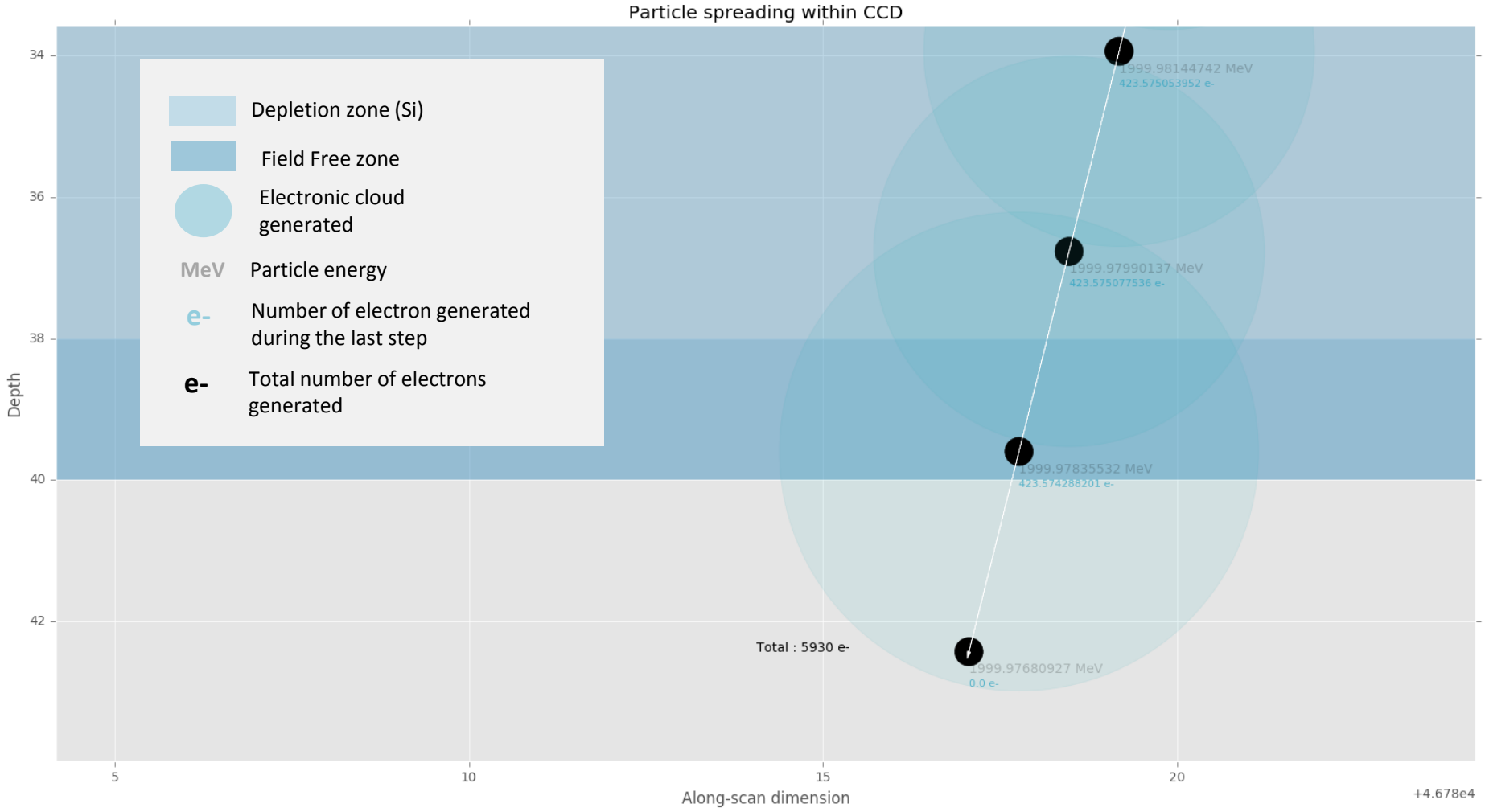
First implementation of T.A.R.S.
(Tools for Astronomical Radiation Simulations)

Example from the simulator with an incoming particle of E_p = 2000 MeV with $\alpha = {}^{\pi}/_{4}$ and a spreading step of $4\ \mu m$



3. Particle spreading

Example from the simulator with an incoming particle of E_p = 2000 MeV with $\alpha = \pi/4$ and a spreading step of $4\ \mu m$



5. Charge diffusion

We have then the expression of the diameter of the initial sigma charge cloud (first cloud created before spreading) :
From Scientific CCD, Janesick

$$D_i = 0.0171[\Delta E]^{1,75}$$

And the final one :
From Hiraga Thesis

$$D_f = \sqrt{z\, dep \frac{k_B\, T}{q\, V}} \sqrt{s + \frac{dep}{z} \ln \frac{dep}{dep - z}} = C \sqrt{s \frac{z}{dep} + \ln \frac{dep}{dep - z}}$$

With

$$C = 1.10^6 \sqrt{\frac{2 k_B\, T\, \varepsilon_{Si}}{N_a q^2}}$$

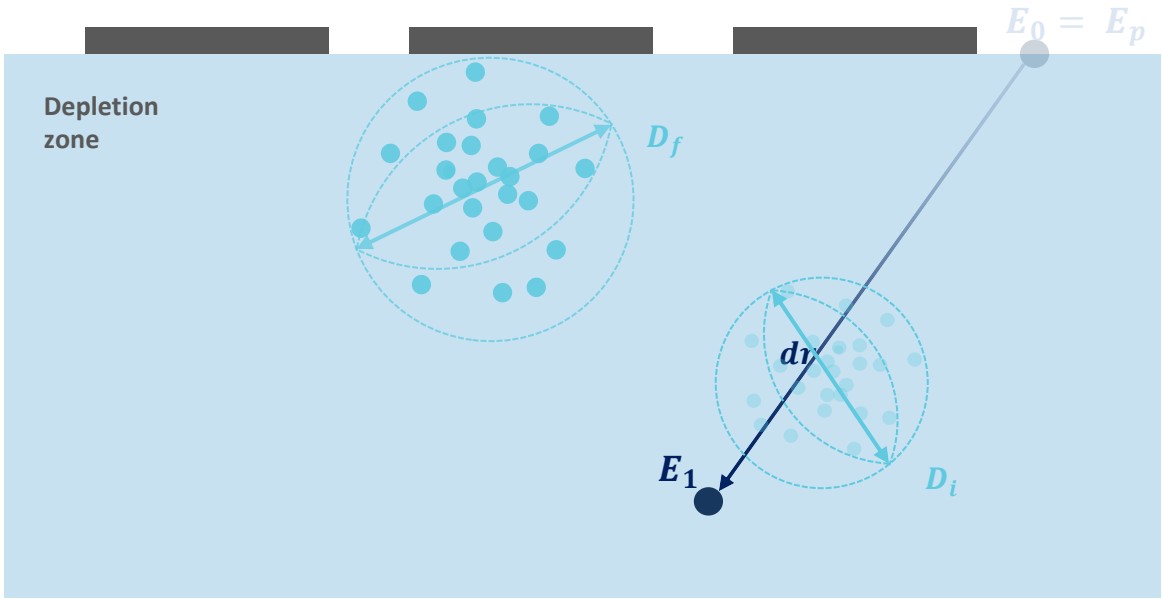
$$s = q\, N_a \frac{dep}{\varepsilon_{Si}\, E_c}$$

where

$$E_c = 1.01\, T^{1.55}$$

and knowing that

$$V = e \frac{N_a}{2\, \varepsilon_{Si}} dep^2$$



Check Hiraga model for more details

5. Charge diffusion

Here is a figure of the diffusion phenomenon extracted From Hiraga Thesis

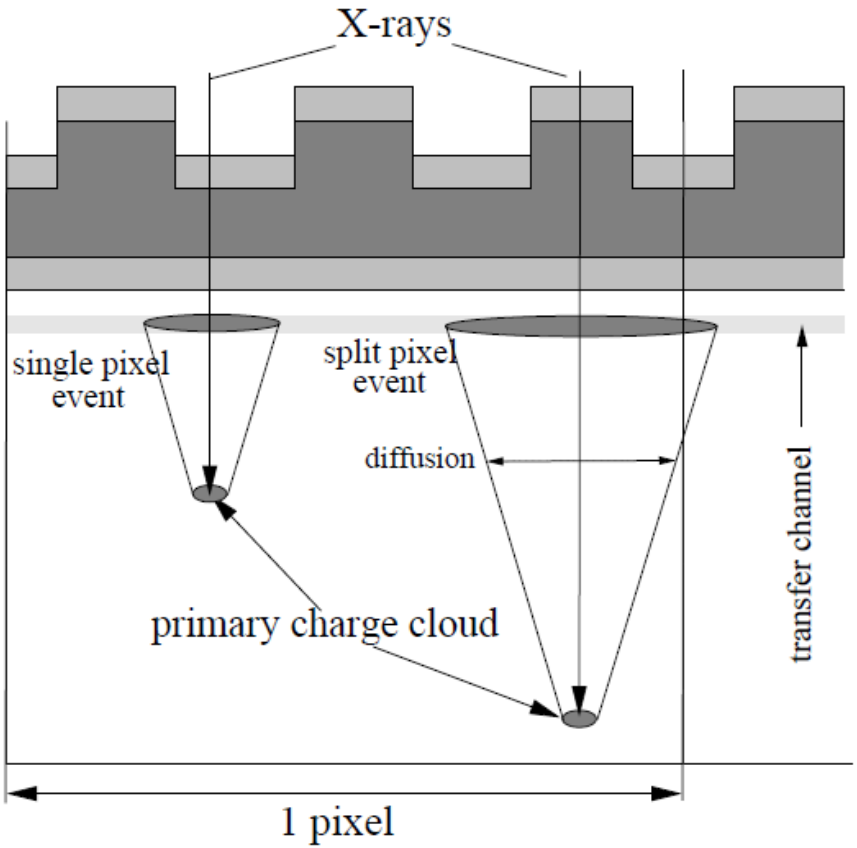
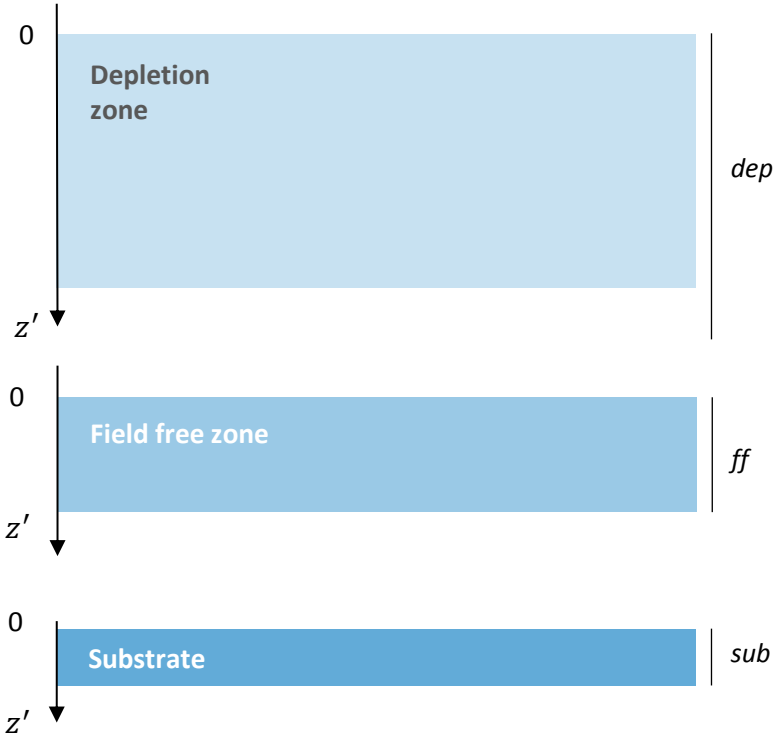


Figure 1.4: A charge cloud generation and its expansion.

5. Charge diffusion

We have the same kind of equations for each layer :



$$D_f = C\sqrt{s\frac{z'}{dep} + \ln\frac{dep}{dep - z}}$$

$$D_f = ff\sqrt{1 - \left(\frac{ff - z'}{ff}\right)^2}$$

$$D_f = \frac{1}{2}\,sub\sqrt{1 - \left(\frac{sub - z'}{sub}\right)^2}$$

Monte Carlo experiments as demonstrated in Fig. 4.47(b) allow one to quantify the diffusion process.[14] It is found that the average lateral dimension from the point of generation gives a two-sigma event diffusion diameter equal to

$$C_{ff} = 2x_{ff}\left(1 - \frac{L_A}{x_{ff}}\right)^{1/2}, \qquad (4.16)$$

where C_{ff} is the lateral diameter of diffusion (cm), x_{ff} is the amount of field-free material (cm) and L_A is the distance from the back surface at which the photon interacts (cm).

6. Charge collection

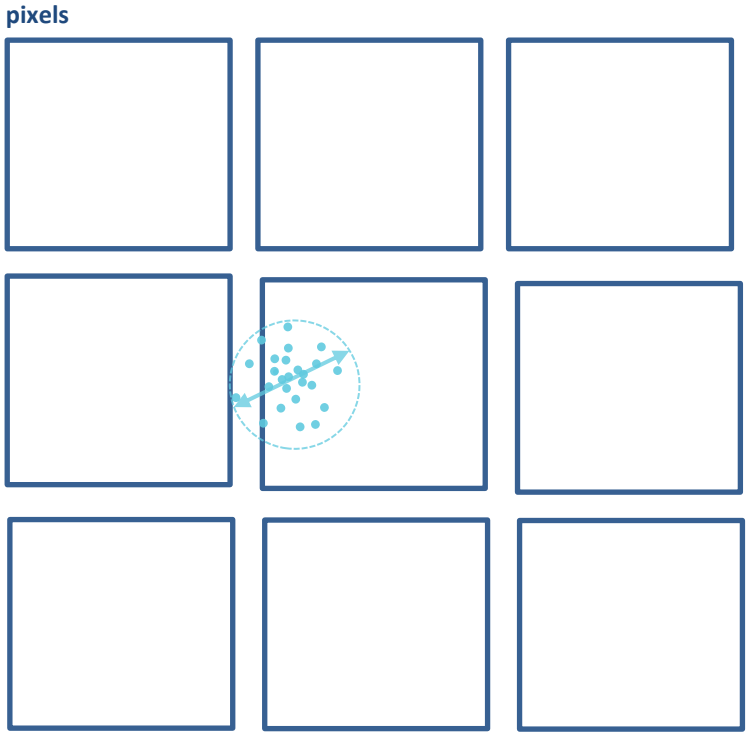
cloud begins to be separated at the pixel borders. The amount of charge that reaches a pixel is given by the integral of the projected charge density over the pixel area. We assume that the charge cloud has a Gaussian radial density profile. Then the ccf is the integral of the charge density over the area of the central pixel plotted for any given photon conversion position in a region of 3×3 pixels. In this case, the ccf is described by a combination of four error functions:

$$\mathrm{ccf}_{\mathrm{mod}}(x;y;\sigma_x;\sigma_y;x_0;x_1;y_0;y_1) = \tfrac{1}{4} \cdot (\mathrm{erf}(x;\sigma_x;x_0) - \mathrm{erf}(x;\sigma_x;x_1)) \cdot (\mathrm{erf}(y;\sigma_y;y_0) - \mathrm{erf}(y;\sigma_y;y_1)). \quad (1)$$

The values σ_x and σ_y give the size of the charge cloud in the row direction and the charge transfer direction, respectively, x_0, x_1, y_0 and y_1 are the coordinates of the pixel borders in the ccf map and the error function $\mathrm{erf}(x,\sigma,x_0)$ is defined as

$$\mathrm{erf}(x;\sigma;x_0) = \frac{2}{\sqrt{\pi}} \cdot \int_0^{(x-x_0)/(\sqrt{2}\sigma)} e^{-t^2}\, dt. \quad (2)$$

See reference [3]



Bibliography

[1] Scientific Charge-Coupled Devices
James R. Janesick

[2] Diagnostics of the X-ray CCD with Subpixel Resolution
Junko HIRAGA

[3] Experimental and theoretical study of the signal electron motion in fully depleted silicon
N. Kimmel a,d, , R.Andritschke

Extraction method (image processing)

Abstract

To study cosmics rays from Gaia FITS we first have to extract those events from images. To do it we are first going to study L.A Cosmics library on different kind of Gaia's data and see if it gives us good enough result to use it safely

Summary

1. L.A. Cosmics library

Quick presentation and example

L.A.Cosmic is a Laplacian Cosmic Ray (CR) Identification Library (written by By Pieter G. van Dokkum (Yale))

You can find everything here
http://www.astro.yale.edu/dokkum/lacosmic/

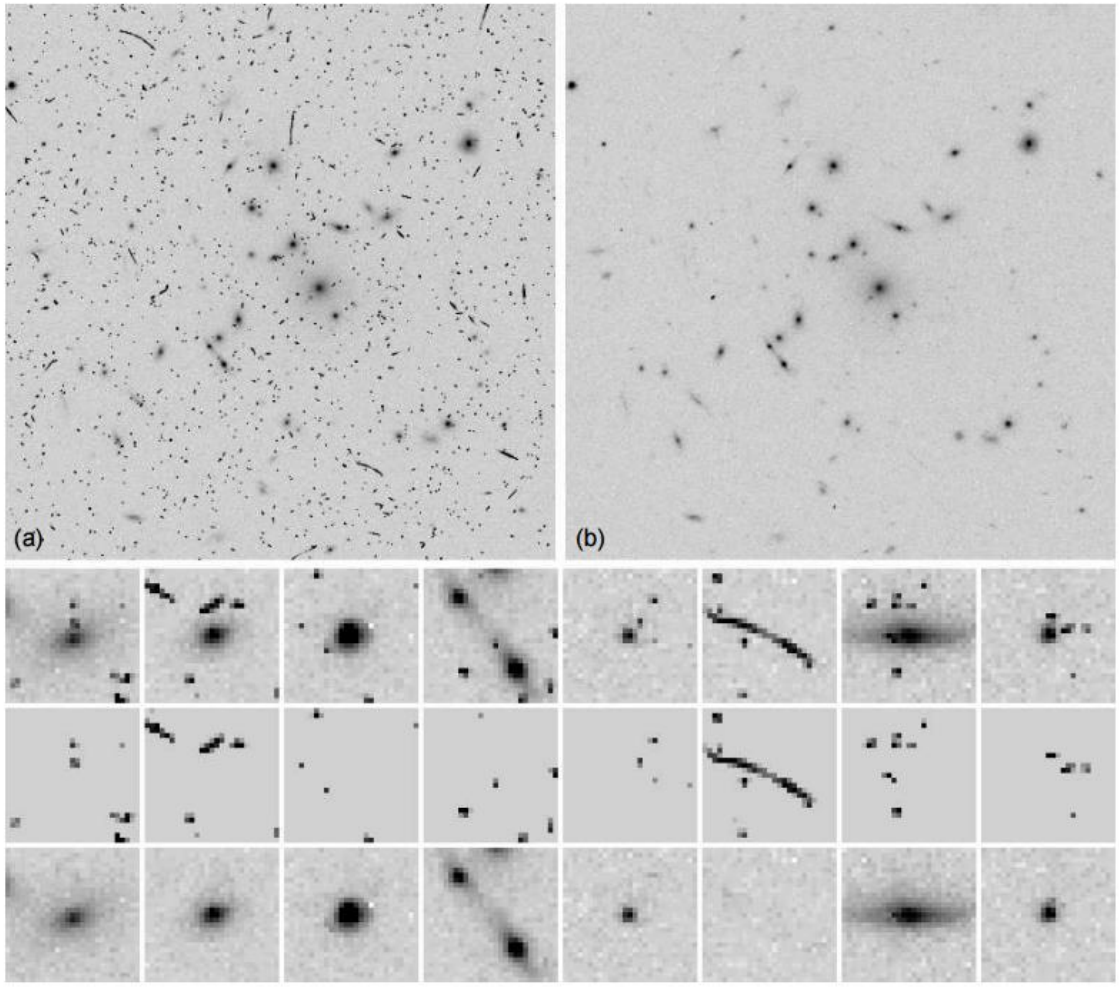
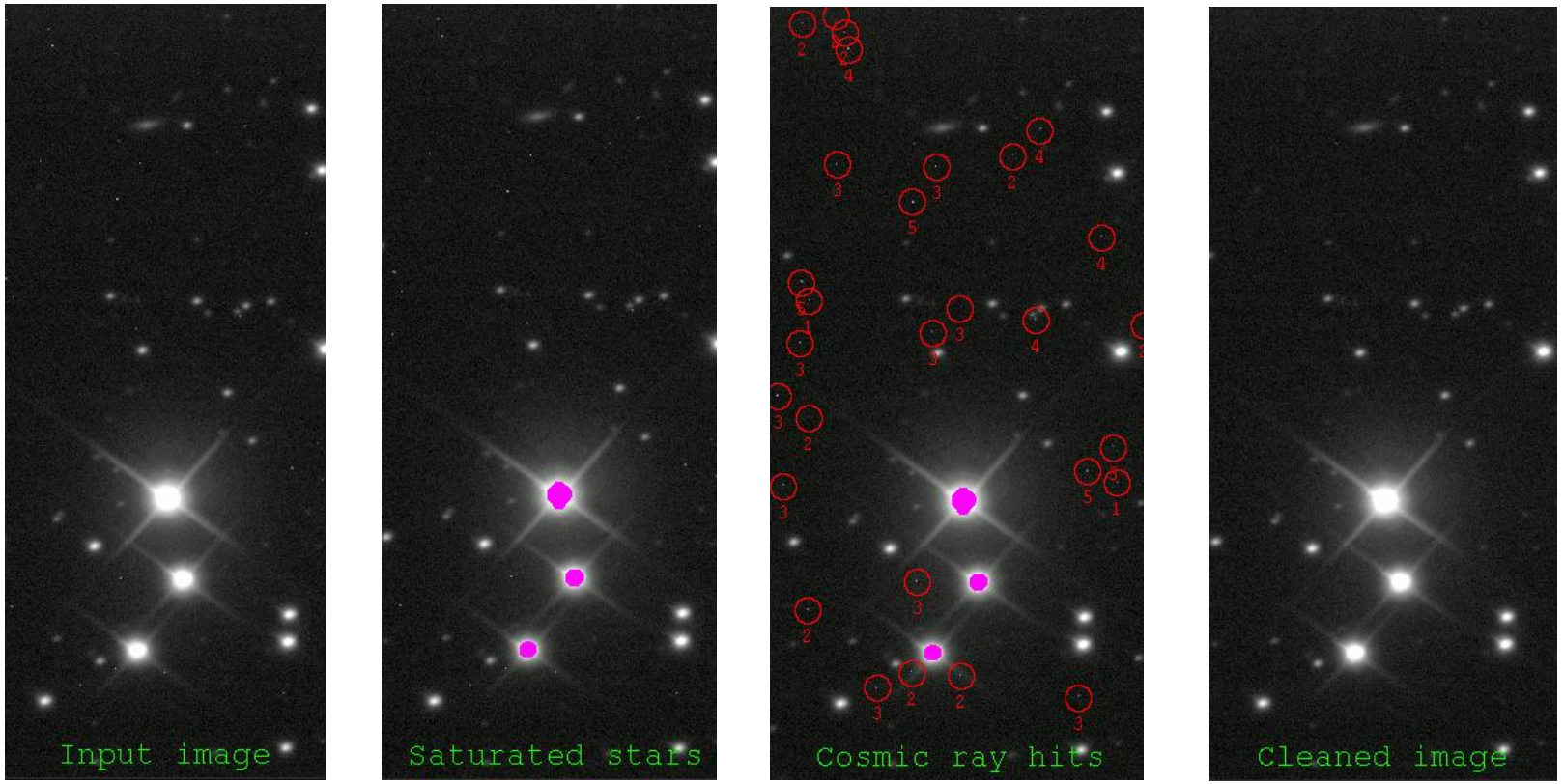


Fig. 6.— (a) HST WFPC2 image of galaxy cluster MS 1137+67. The restoration by L.A.COSMIC is shown in (b). Small panels show close-ups for a selection of stars and galaxies in various WFPC2 images. The algorithm leaves stars intact, and is able to remove cosmic-rays of arbitrary shapes and sizes.

1. L.A. Cosmics library

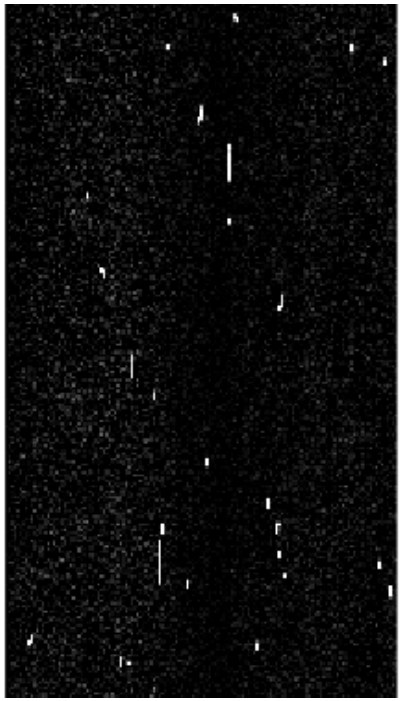
Quick presentation and example

LA cosmics identify saturated stars first to avoid their detection as cosmic rays

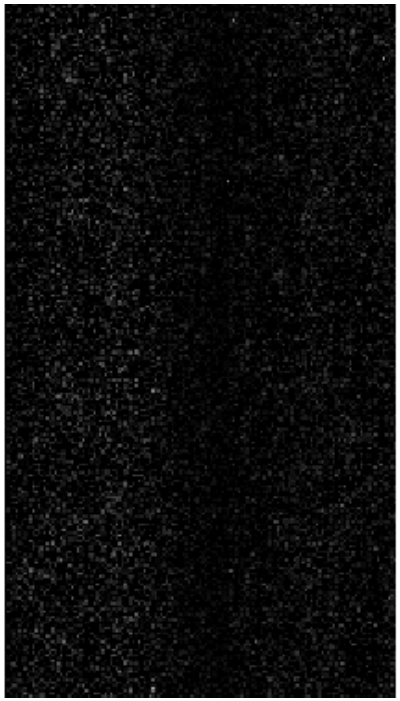


1. L.A. Cosmics library

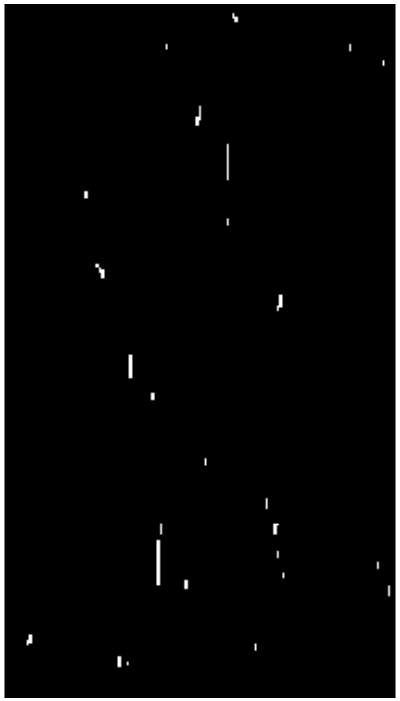
Here is an example applied to a Gaia BAM fits, without stars (We'll study other CCDs data, containing stars later)



Original CCD
image from Gaia



Processed image



Extracted Cosmic
Rays

2. Extraction process

We are first going to describe our method to extract cosmic rays considering the background of the image and other parameters

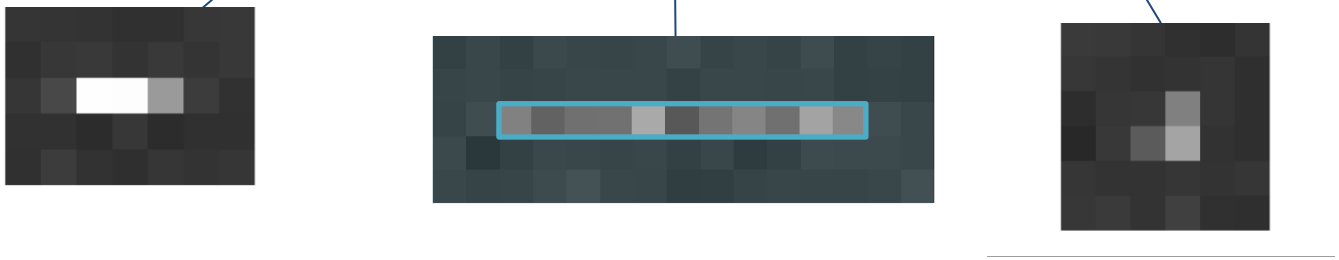
Then we will test this method and try to know if LA cosmic is really an efficient extraction algorithm. Our main goal is to extract every single cosmic rays from the image without changing their shape and value (in number of electrons generated)

2. Extraction process description

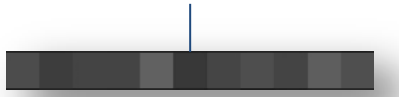
Zoom on a Gaia BAM Image (with 3 CRs)

First, we extract the cosmic rays using L.A. cosmics library (here is an example of 3 cosmic rays on a BAM image, without stars,. Only the mask is generated by L.A. cosmics is important)

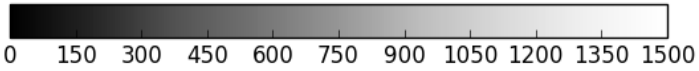
Mask generated with L.A. cosmics



Then, we subtract the local **mean** to evaluate the number of electrons generated by the CR event



We do not forget the *gain factor* !!!



Number of electrons

2.1. Extraction process validation (on BAM)

Here is our procedure to validate our extraction method using LA cosmics :

1. We **clean an image of Gaia** with L.A. Cosmics (Reference image)
   (here we are going to do it on a BAM CCD to avoid the detection of stars. We will test this other case later)

2. We **add simulated Cosmic Ray** events on this clean image
   (In a first version of this test, simulated cosmic rays are coming from a catalog generated some times ago with the Fortran version simulator – catalog_2)

3. We use L.A. Cosmics to **clean the simulated image**

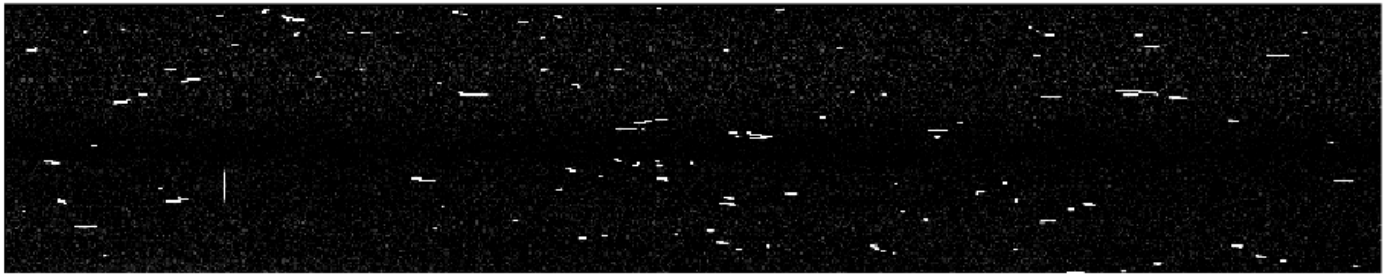
4. Statistical study of the **extraction performance**
   (First we are just going to test if the number of extracted cosmic rays is the same as the number of injected cosmic rays. On the 2. step, we inject exactly the same number of cosmic rays extracted at step 1. from the real image)

(results on next slide)

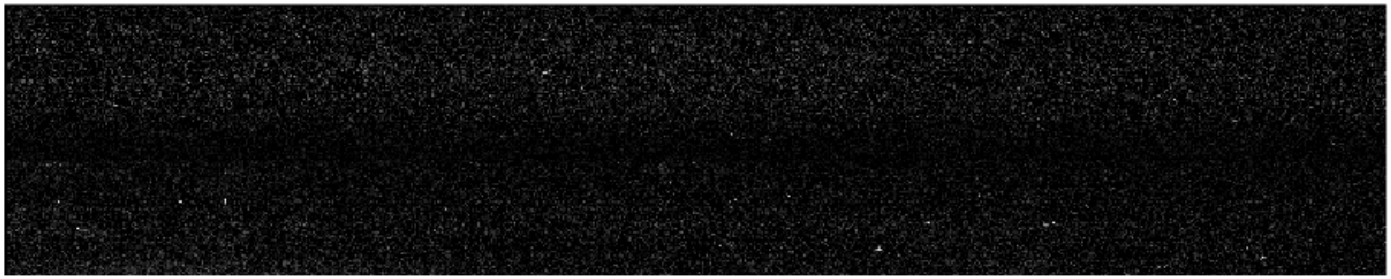
2.1. Extraction process validation (on BAM)

Extraction rate:

99.1% (BAM data)
of injected cosmic rays have been correctly extracted



Simulated Gaia BAM **CCD Image**



Cleaned image

2.1. Extraction process validation (on BAM)

Now we know that we can use LA cosmics BAM CCD data from Gaia to extract approximately 99% of the cosmic rays on the image

But now we want to know if our complete extraction method is able to give us complete cosmic rays, i.e. extracting exactly the number of electrons produced by the cosmic ray. Here is a method to do it :

1. We **clean an image of Gaia** with L.A. Cosmics (Reference image)
   (here we are going to do it on a BAM CCD to avoid the detection of stars. We will test this other case later)

2. We **add simulated Cosmic Ray** events on this clean image
   (In a first version of this test, simulated cosmic rays are coming from a catalog generated some times ago with the Fortran version simulator – catalog_2)

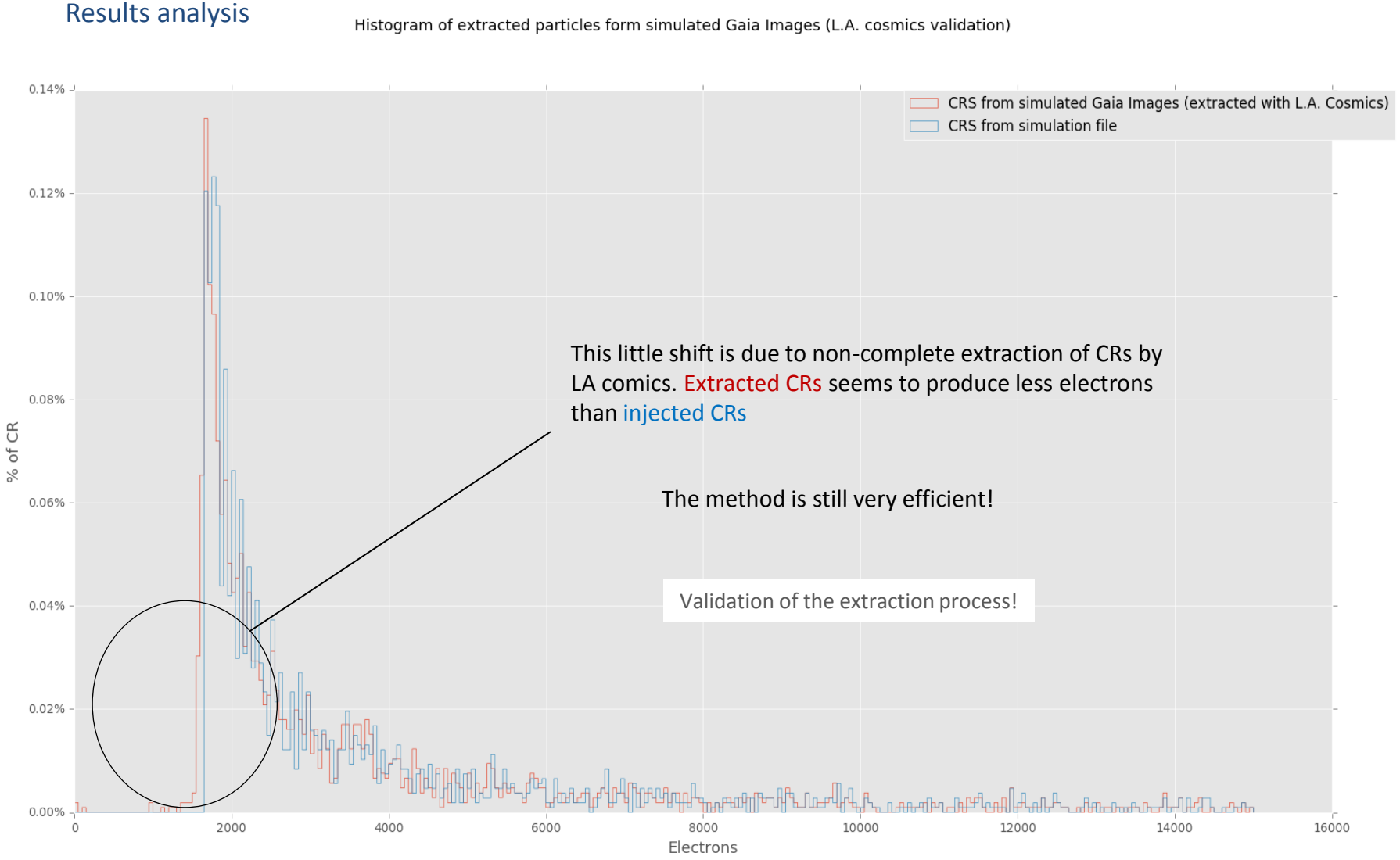
3. We use L.A. Cosmics to **extract the cosmic rays and create a catalog** containing all the single cosmic rays with the number of electrons generated/extracted (For more details about the catalog, see the section : Developing tools to do an efficient statistical study of CRs)

4. Statistical study of the **extraction performance**
   (We are going to do an histogram of the number of produced electrons for both the injected CRs and the extracted CRs. If our method is efficient, histograms should be the same)

(results on next slide)

2.1. Extraction process validation (on BAM)

Results analysis

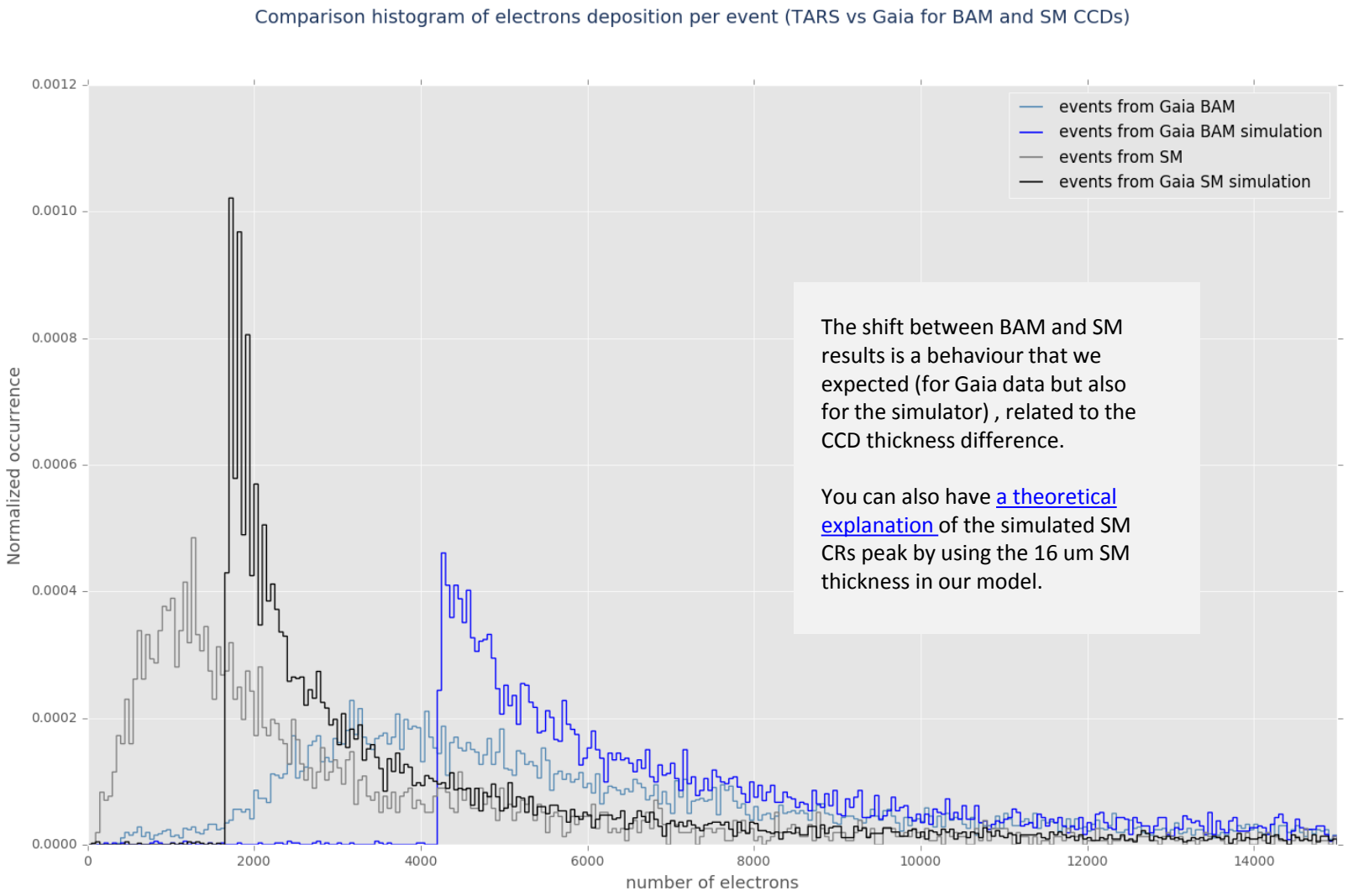


This little shift is due to non-complete extraction of CRs by LA comics. Extracted CRs seems to produce less electrons than injected CRs

The method is still very efficient!

Validation of the extraction process!

3. Extraction Results



The shift between BAM and SM results is a behaviour that we expected (for Gaia data but also for the simulator) , related to the CCD thickness difference.

You can also have a theoretical explanation of the simulated SM CRs peak by using the 16 um SM thickness in our model.

Fortran to Python

Abstract

The CR simulator we are using is written in Fortran. To use this simulator in the context of other modern astronomical applications and tools, we want to adapt this code into a Python code. Then we could use this simulator as a basis of a complete simulator for CCDs studies including other kind of physical simulation (CTI, radiations…etc.) into a same interface.

Summary

First we are going to describe in details the inputs of this CRs simulator
To see the Python implementation details see …
To see the Physical model details see…

Then we will study all the performances of the two simulators. One written in Fortran, the other one written in python.

1. Simulator inputs

A more physical version of this part can be found here

There is different types of physical inputs :

- configuration files (cfg) containing simulation and CCD specs
- Physical parameters in a txt file

Both are read by the simulator and translated into variables and useful functions

CCD CONFIG	
number_of_pixels_ac	Number of pixels in the across-scan direction
number_of_pixels_al	Number of pixels in the along-scan direction
temperature	CCD temperature (K)
depletion_zone_thickness	Depletion zone thickness (μm)
field-free_zone_thickness	Field free zone thickness (μm)
substrate_thickness	Substrate thickness (μm)
pixel_ac_size	Pixel across-scan size (μm)
pixel_al_size	Pixel along-scan size (μm)
electrons_saturation	Electrons saturation (number of $e-$)

CCD configuration file

1. Simulator inputs

SIMULATION CONFIG		
number_of_particles	*integer*	Total number of incident particle on CCD
energy	*"random"* *float* *range()* *np.logspace()*	Energies of the particles
position_x	*float*	across_scan positions of the particles (μm)
position_y	*float*	along_scan positions of the particles (μm)
alpha_angle	*float*	alpha incident angle of the particles (rad)
beta_angle	*float*	beta incident angle of the particles (rad)
spreading_step	*float*	spreading step in CCD
INPUTS FILE		
input_file	*['yes'/'no']*	input file containing a pre-computed spreading of the particule
positions	*file_path (string)*	file of positions pre-computed
energies	*file_path (string)*	file of energies pre-computed
SIMULATION REPORT		
date	*year-month-day*	date of the simulation - written after simulation
processing_time	*hours-minutes-seconds*	processing time of the simulation - written after simulation

Simulation configuration file

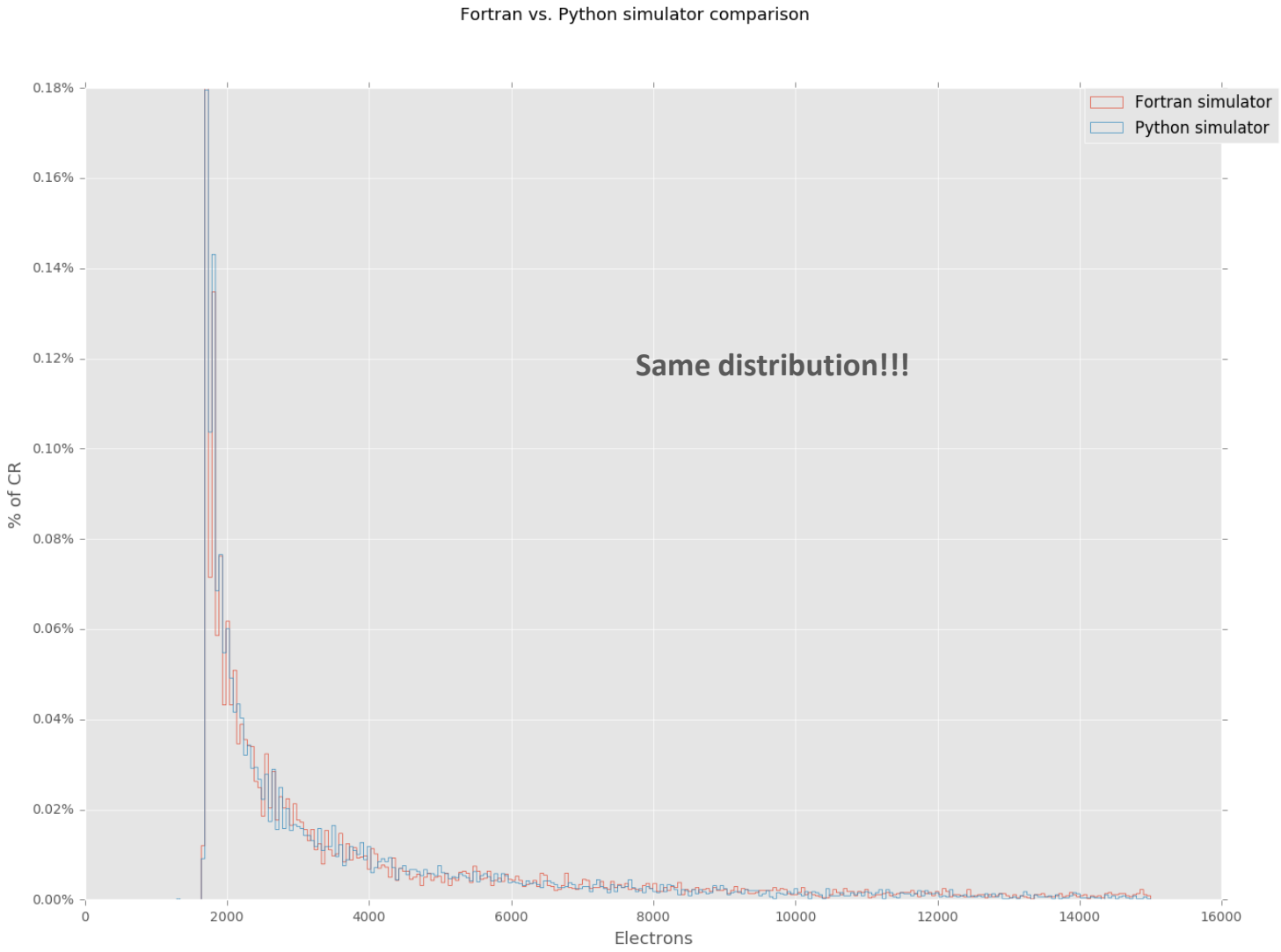
2. Python vs. Fortran performance comparison

We will study in this part all the performances of the two simulator. One written in Fortran, the original one.
The other one written in python

First we are going to study the efficiency of python code to reproduce the Fortran simulated data. We are going to use an histogram of generated electrons to do it and to compare the energetic distribution of simulated CRs. We are also going to compare all the CR characteristics defined here from the two simulators

Then we are going to compare the simulation time of the two simulator.
(of course we expect Fortran code to be the faster but we will see later all the benefits of a good python library. We will develop this point on the General simulation tools section)

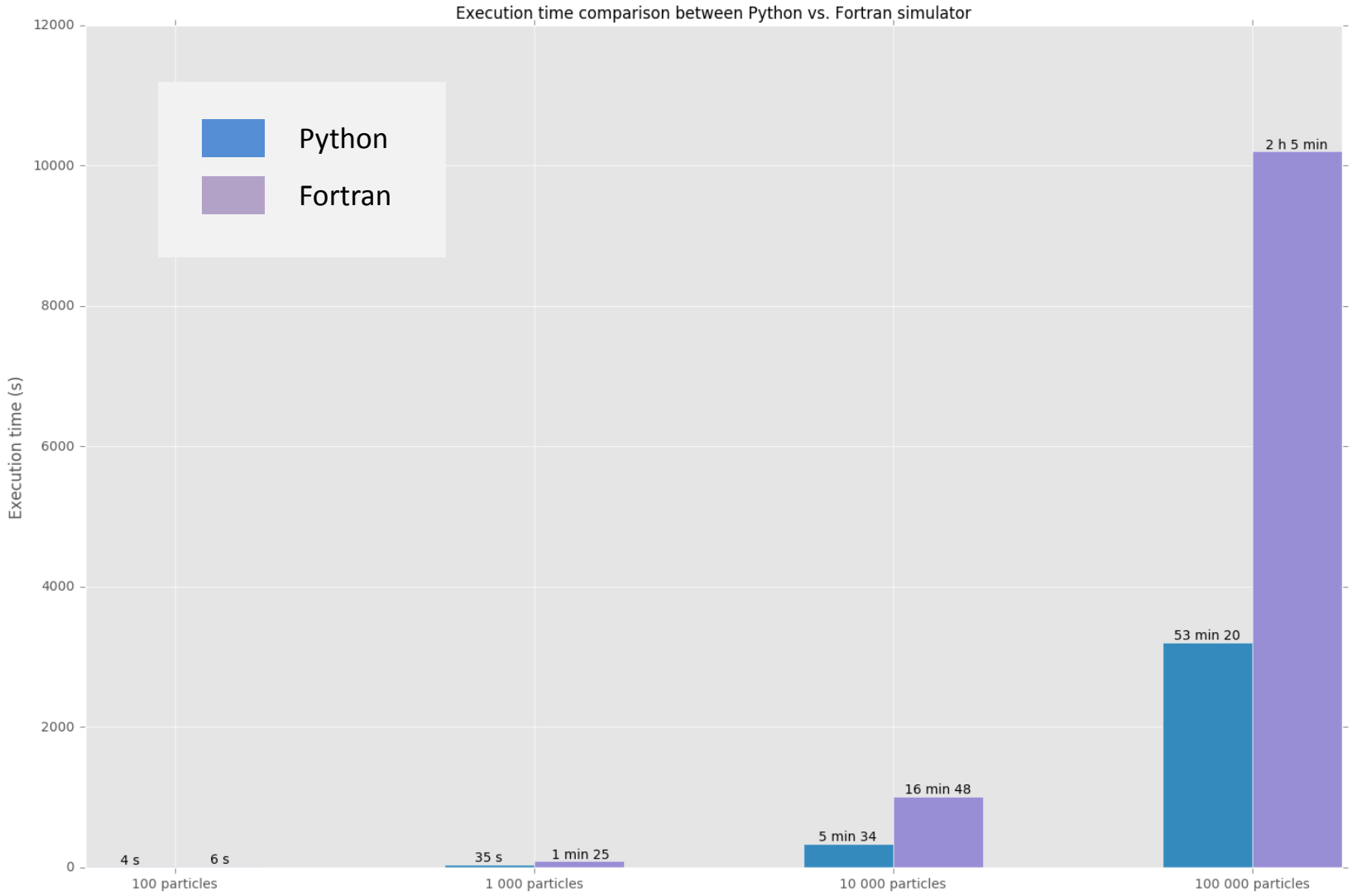
2.1. Correspondence test between simulated data – Results 1



This test has been made with 10 000 simulated CRs from both simulators with the same inputs parameters

2.2. Simulation time comparison

Simulation on a x64 OS (Windows), 8GB RAM and CPU @2.30 GHz



Parameters influence and problems with the first implementation (BAM)

Abstract

This part contain a study of the simulation parameters influence on result accuracy

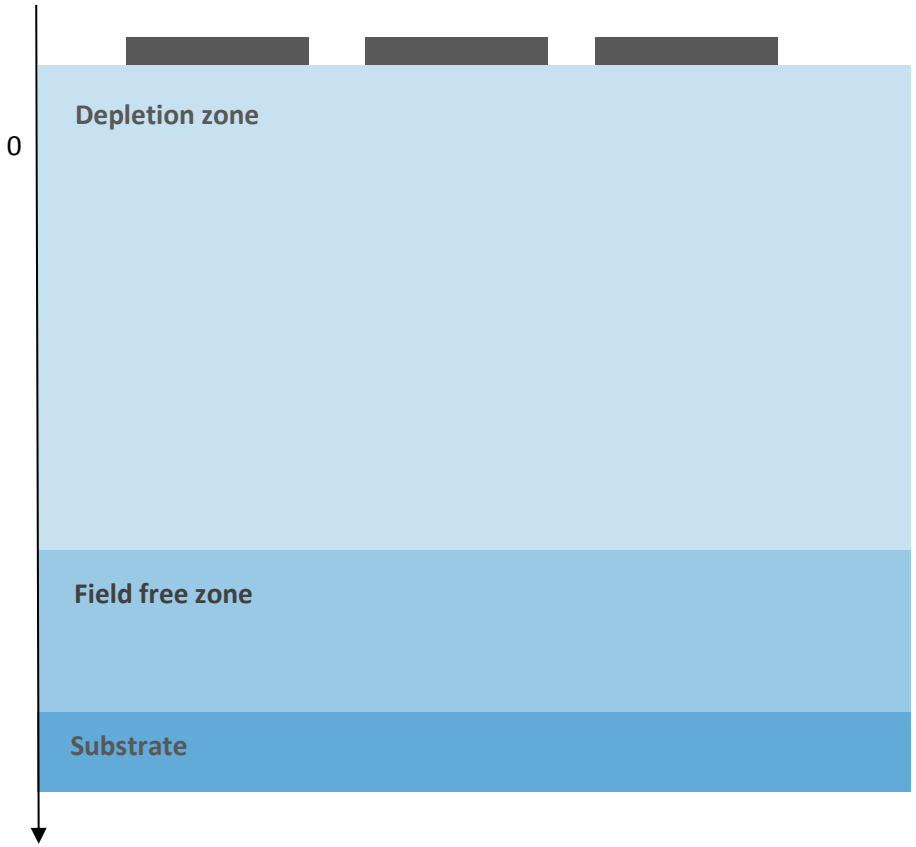
Summary

1. Inputs for Gaia (from original implementation)
2. First problem : 4250 MeV limit explanation
3. Second problem : Spreading step parameter influence

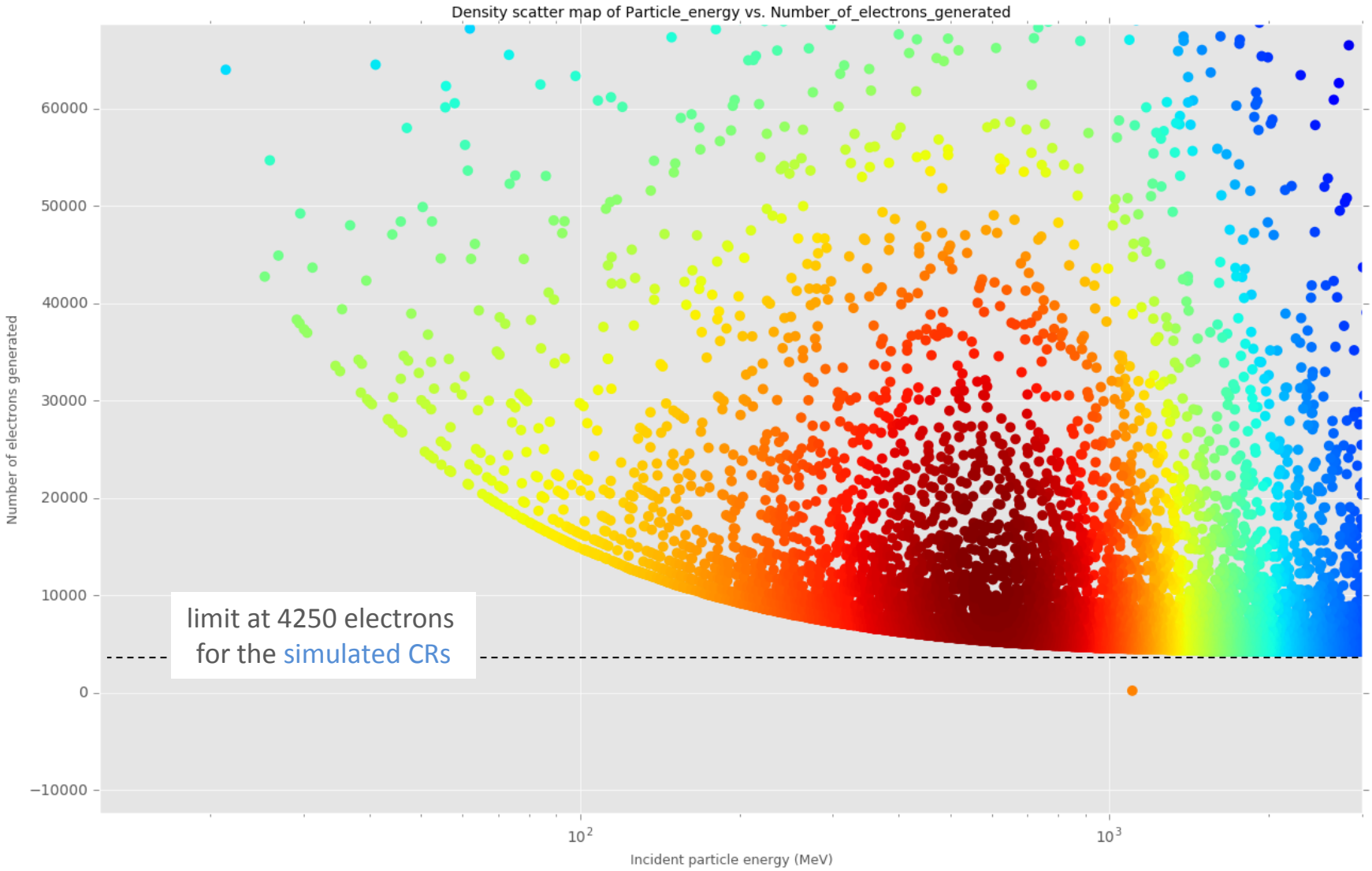
1. Inputs for Gaia (from original implementation)

CCD Structure :

- Temperature : 163 K
- Field free thickness : 2 μm
- Total CCD thickness : 40 μm
- Substrate thickness : ~ 0 μm
- Shielding : 11 mm aluminium
- Pixels-dimension : 30 μm x 10 μm



2. First problem : 4250 MeV limit explanation



2. First problem : 4250 MeV limit explanation

Let's try to understand what the problem by finding the minimal number of electrons that a CR generated by our generator can generate.

All the equations come from the model

The number of electrons is

$$N(e-) = \frac{\Delta E}{E_{e-h}}$$

with

$$\Delta E = 100\, S_m(E_0)\, n_{Si}$$

So

$$\min(N(e-)) = \frac{100\, \min(S_m(E_0))\, n_{Si}}{E_{e-h}}$$

We have in our inputs :

$$\min(S_m(E_0)) = 1.661$$

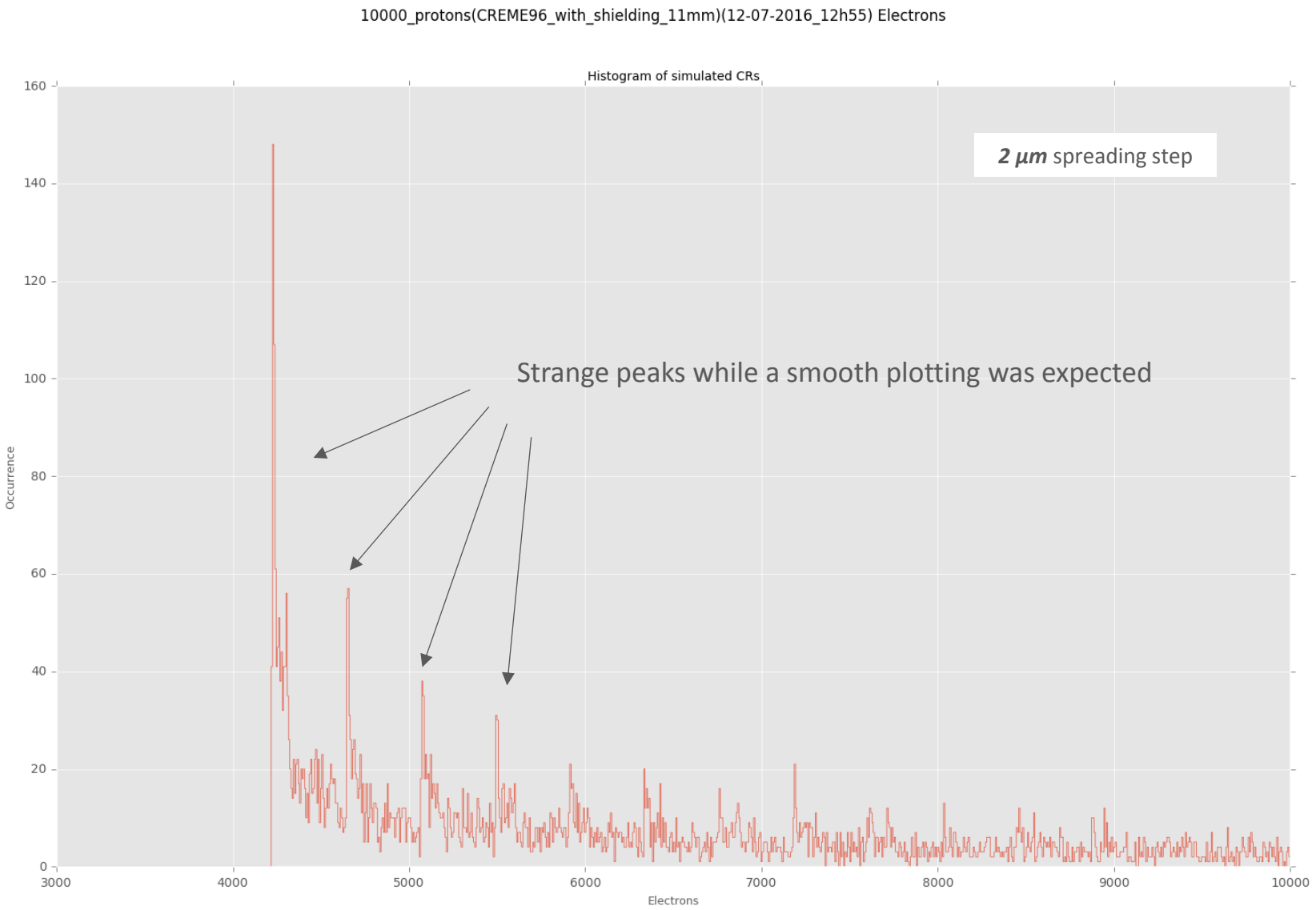
So

$$\min\left(N(e^-)\right) = 106\, e^-$$

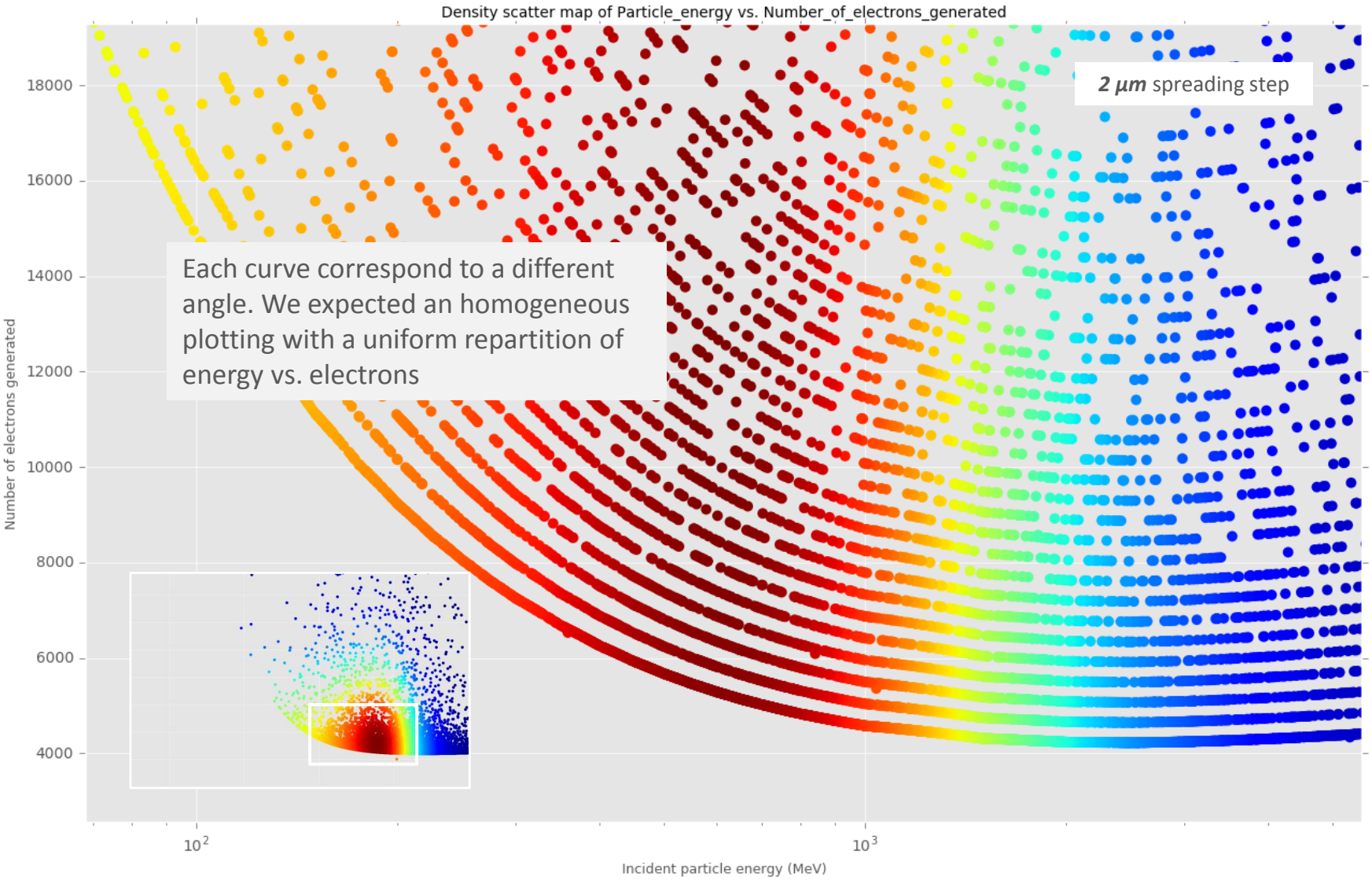
Let's suppose now that the particle spreads into all the CCD (that's what is happening most of the time). To stay in minimum values, let's consider a normal incident particle crossing 40 um of CCD (with $40\mu m$ steps of $1\mu m$) :

$$\min\left(Total\ N(e^-)\right) = 106\, e^- * 40 = 4240\, e^-$$

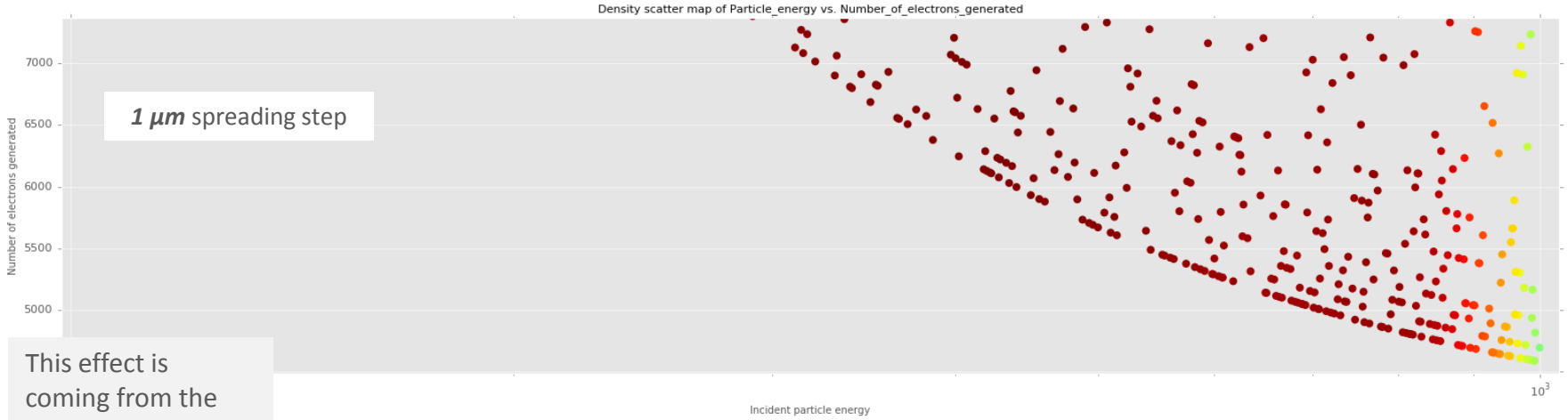
3. Second problem : Spreading step parameter influence



3. Second problem : Spreading step parameter influence

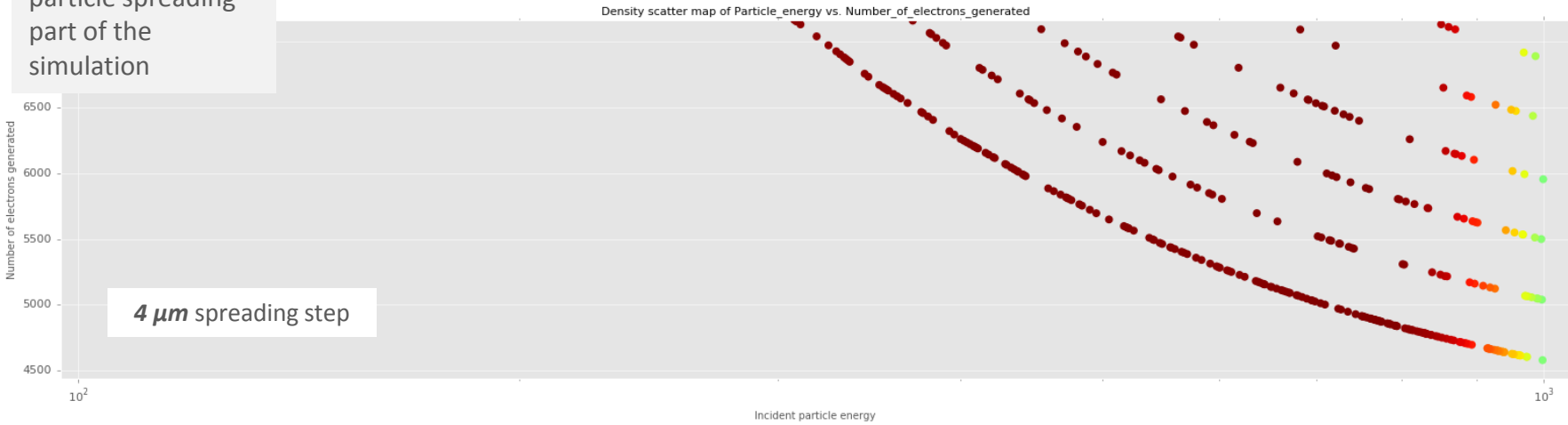


3. Second problem : Spreading step parameter influence



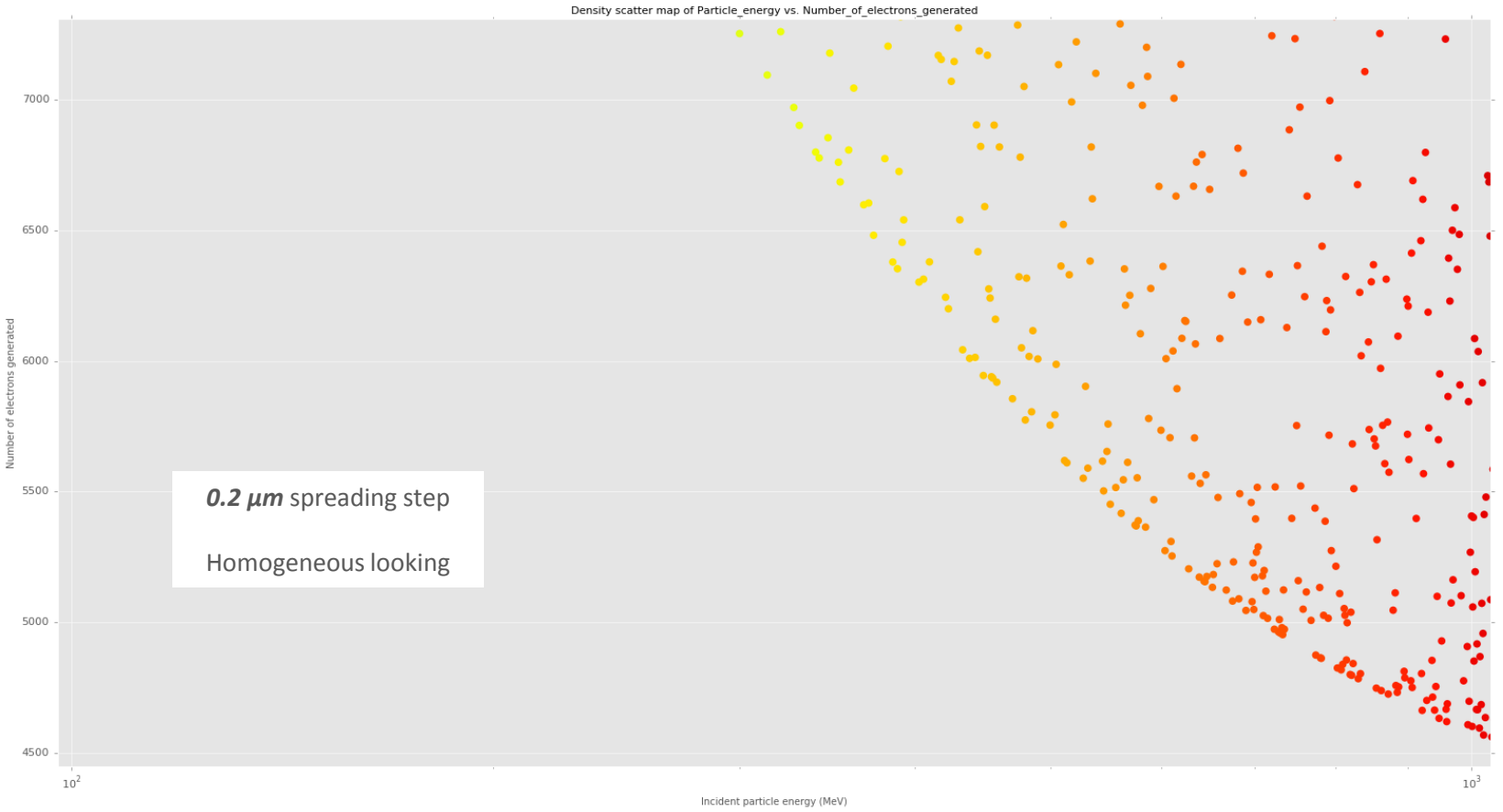
1 µm spreading step

This effect is coming from the spreading step we have chosen in the particle spreading part of the simulation

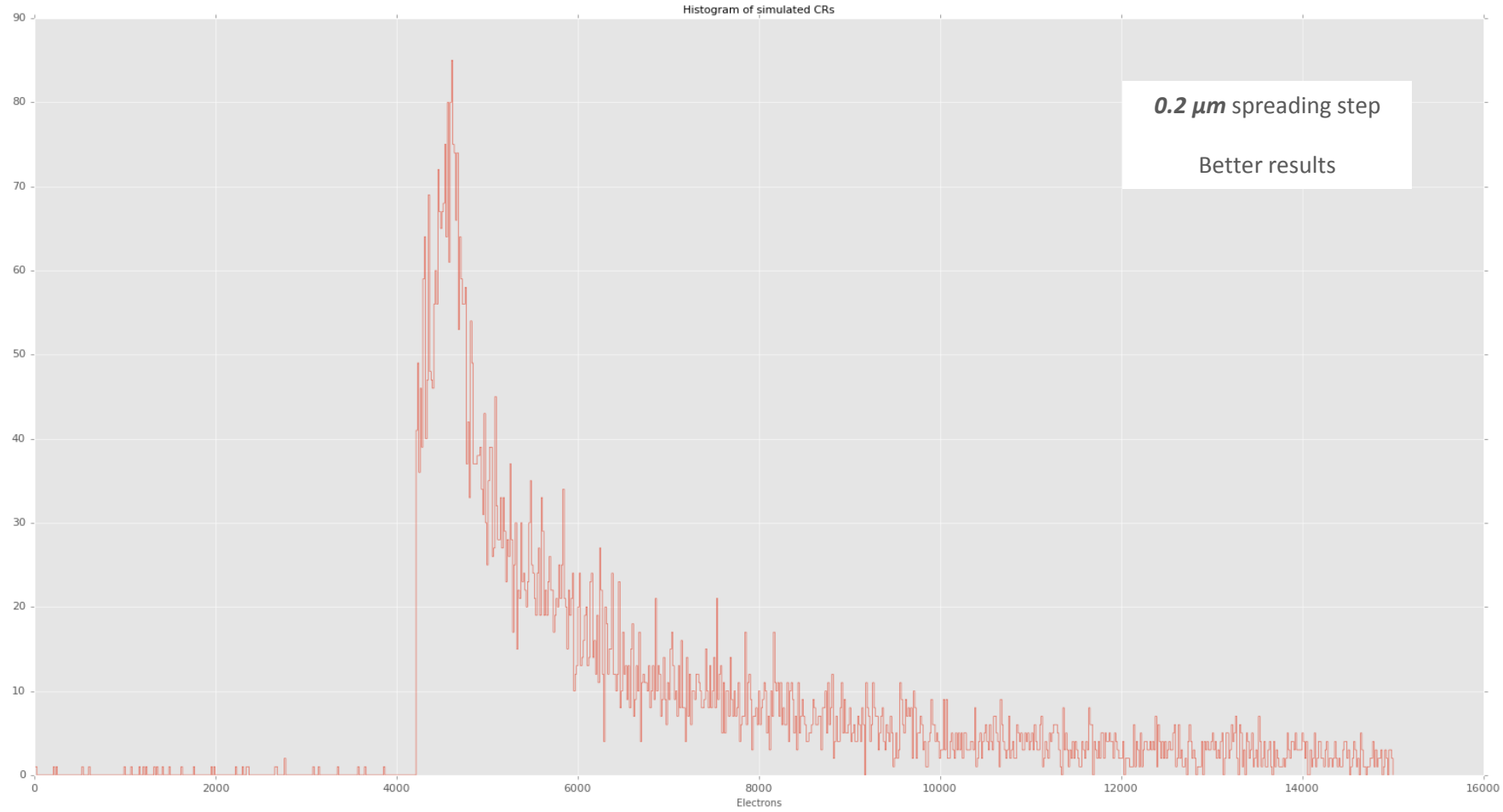


4 µm spreading step

3. Second problem : Spreading step parameter influence



3. Second problem : Spreading step parameter influence



Geant4 results

Abstract

This part contain a study of the Geant4 simulation via the ROOT output. We are mainly going to study the GCR incoming protons and helions simulated for Gaia.

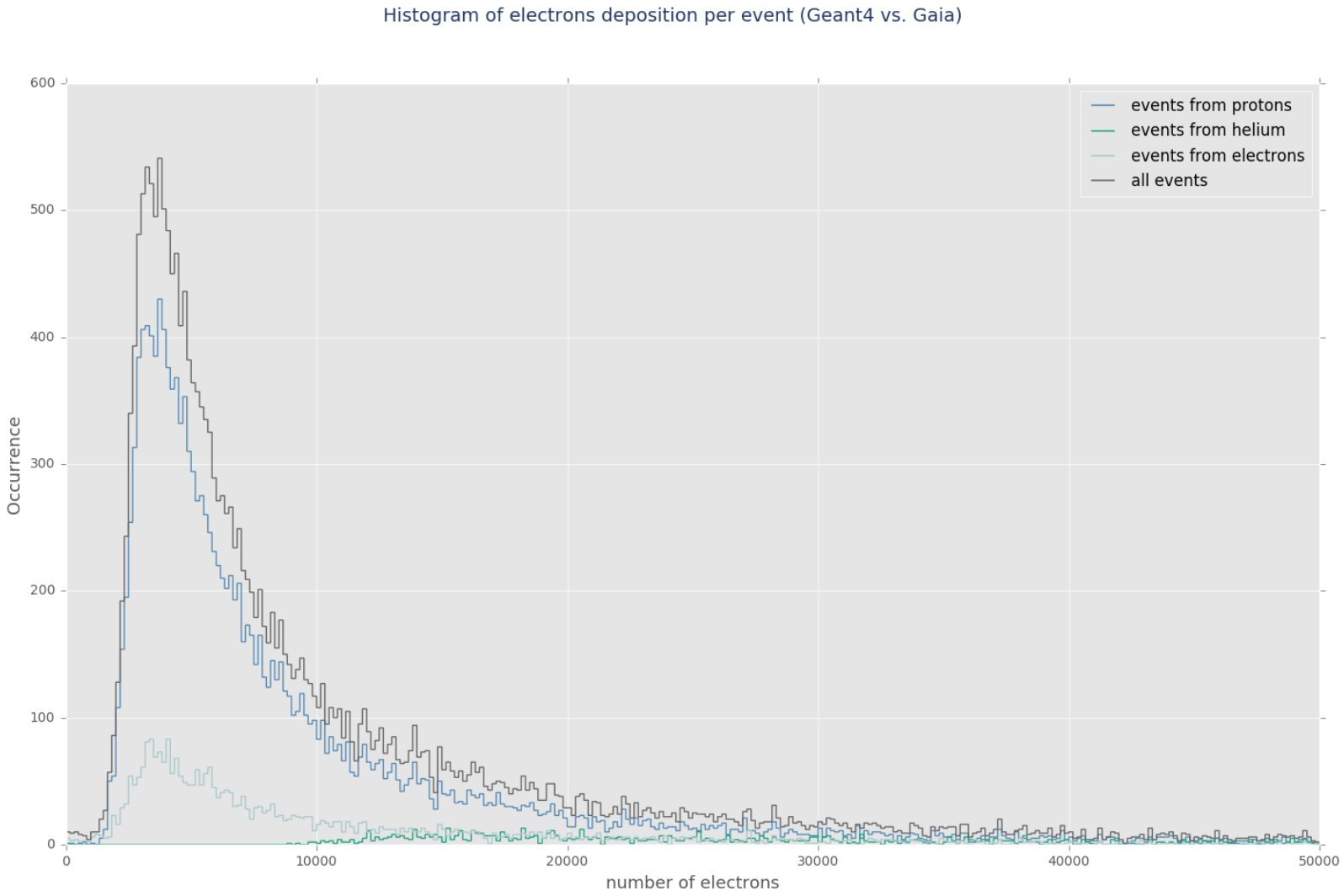
Summary

Histogram comparison of particles contribution to e- generation

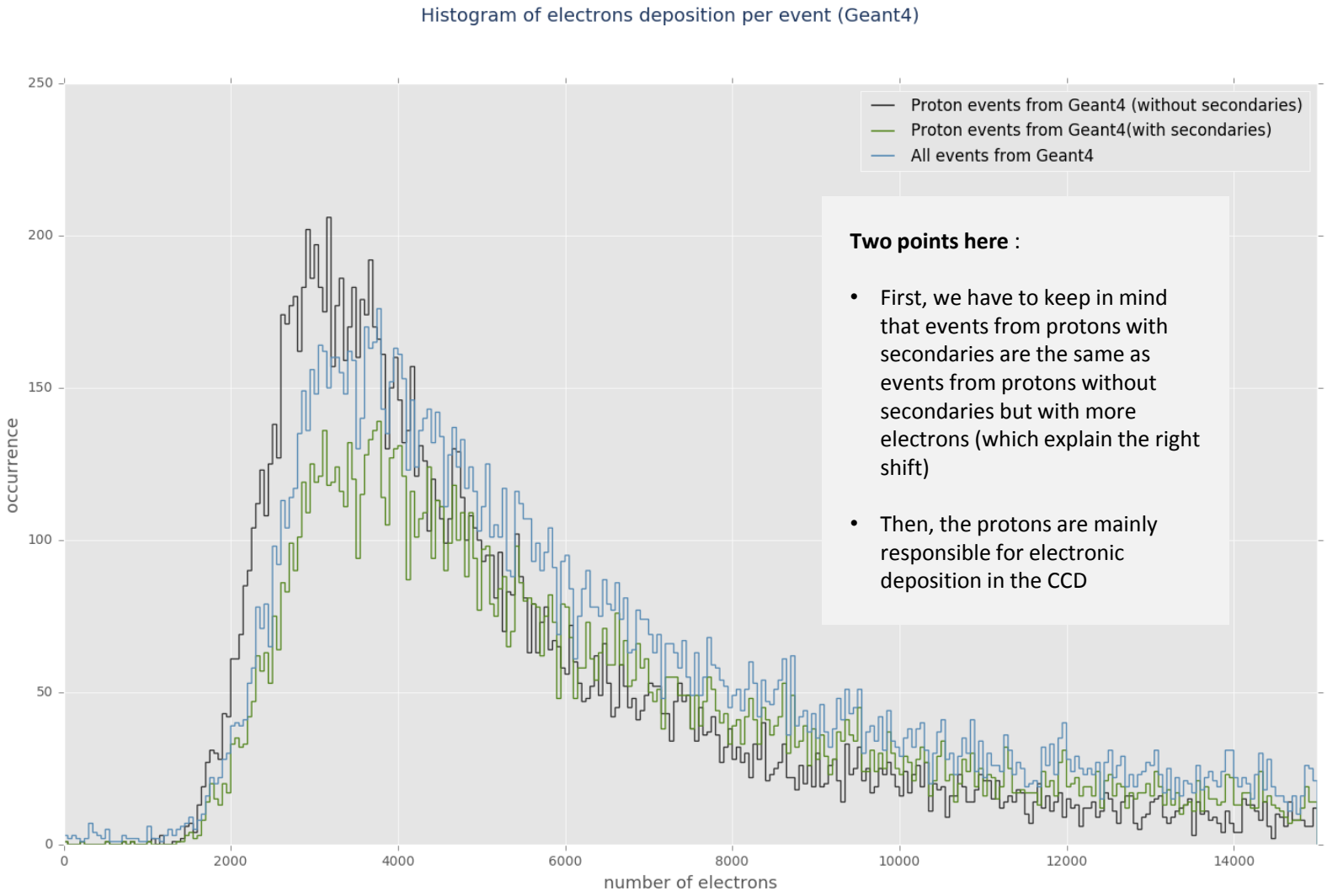
Histogram comparison to understand protons importance on CRs events

Particles proportions at the BAM CCD Surface (from Geant4 simulation)

Histogram comparison of particles contribution to e- generation



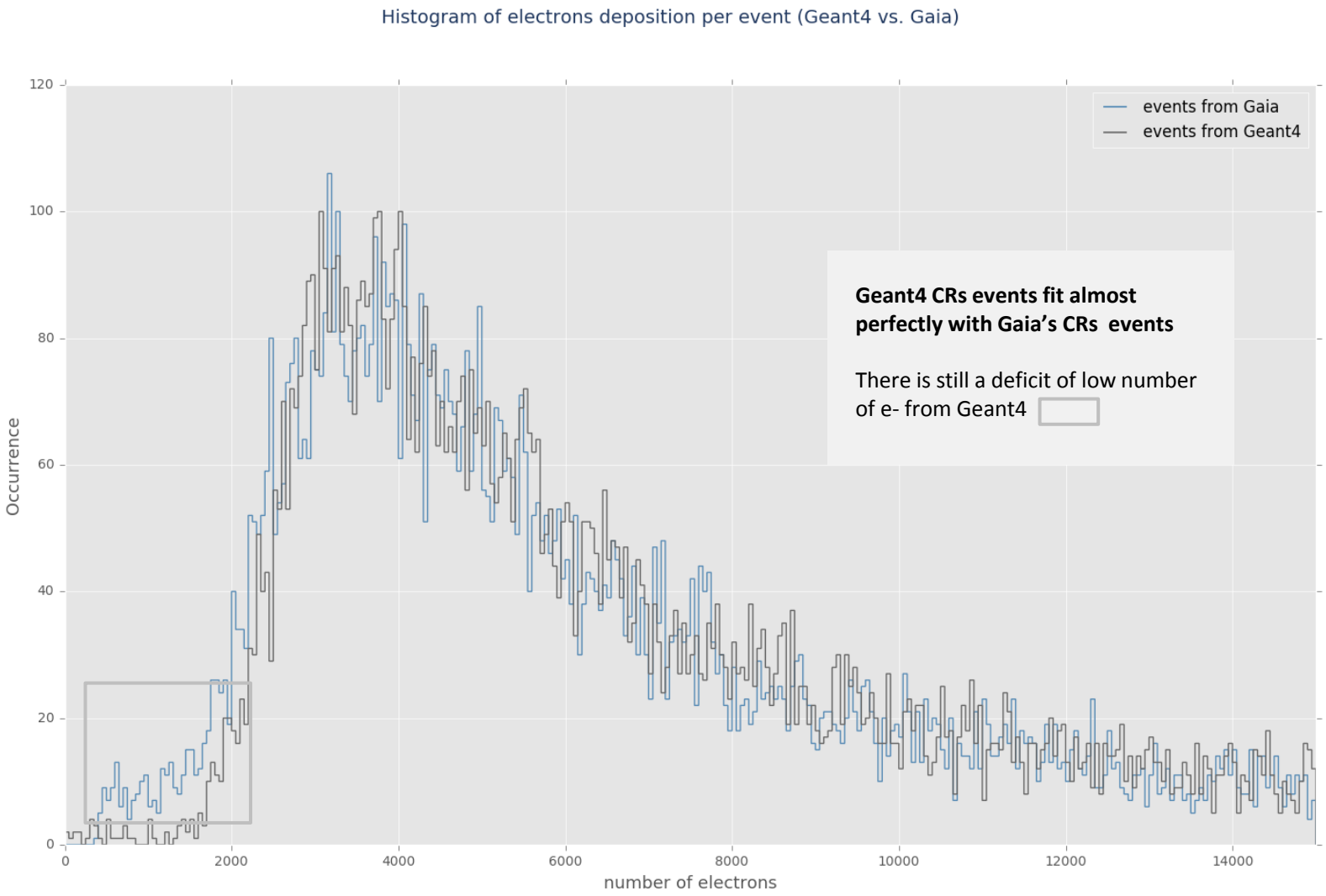
Histogram comparison to understand protons importance on CRs events



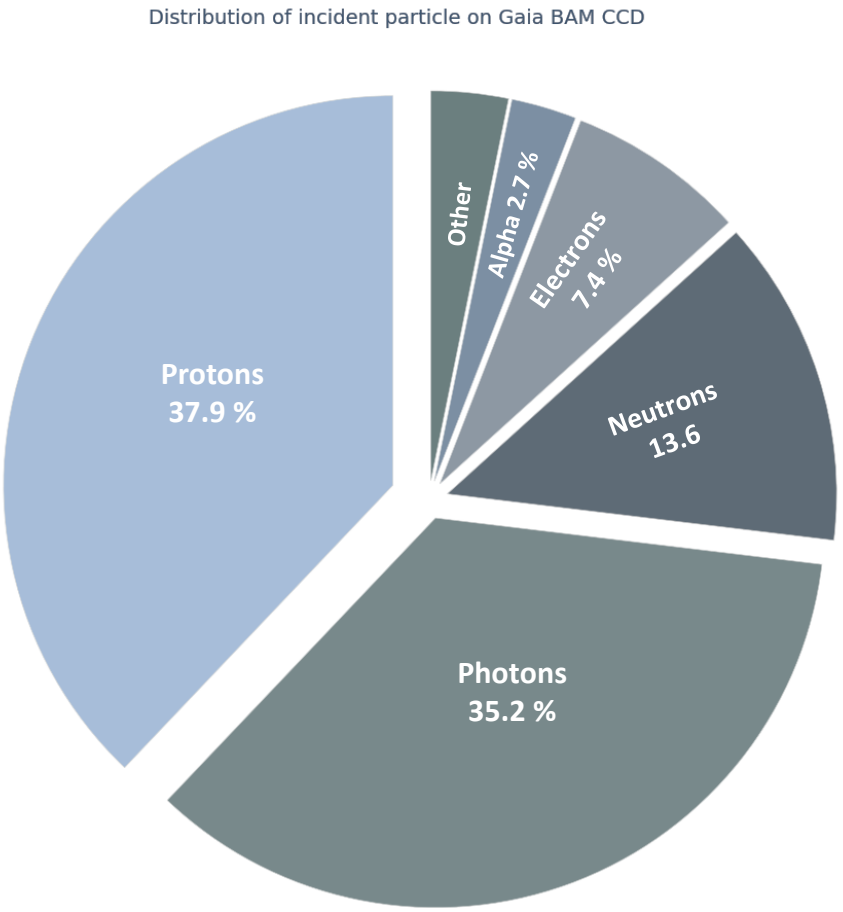
Two points here :

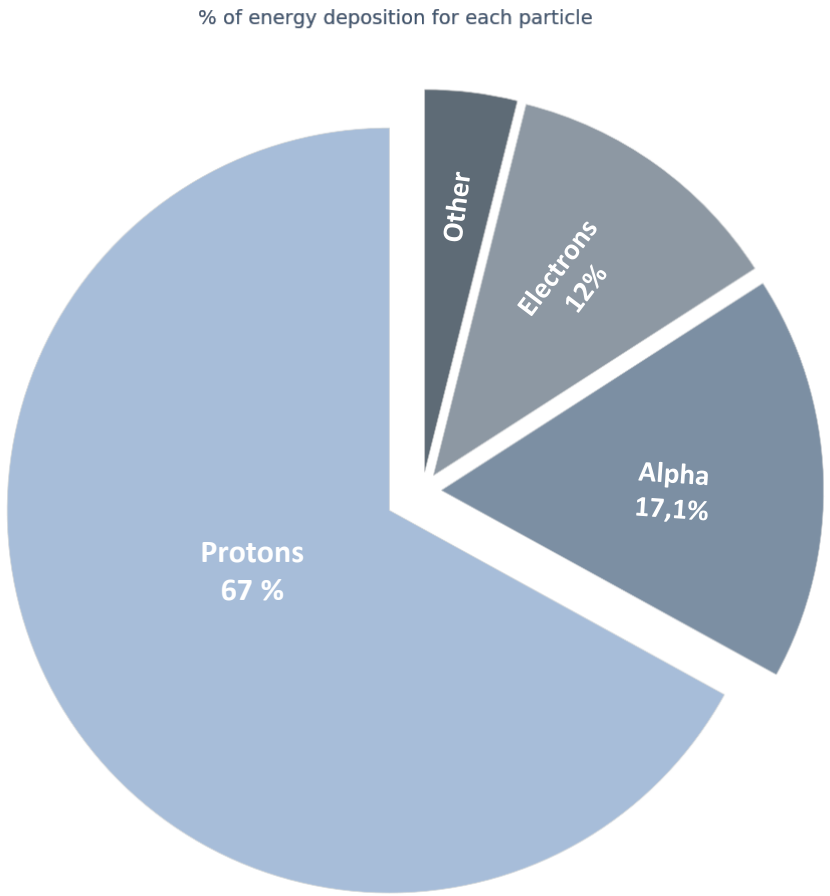
- First, we have to keep in mind that events from protons with secondaries are the same as events from protons without secondaries but with more electrons (which explain the right shift)
- Then, the protons are mainly responsible for electronic deposition in the CCD

Histogram comparison to understand protons importance on CRs events



Particles proportions at the BAM CCD Surface (from Geant4 simulation)





Possible improvements and analysis
(personal point of view only)

Complete analysis of interplanetary electrons and solar protons from Geant4. They could explain the deficit of low electrons events with Geant4 (for now the study has been made only for GCR protons)(see this plot)

For now the number of particles simulated for one image is arbitrary. Discuss with people from TEC-EES on how to use Geant4 simulation to properly use the incoming particle flux and know exactly what **the real amount of GCR particles for one BAM CCD FITS** (depending on the exposition time) (Geant4 normalise the Monte-Carlo simulation results. Marco Vuolo proposed me to help with this normalisation problem)

Early stage testing problem : During the early development phase we don't have a 3D model of the spacecraft. Still we can use Geant4 to simulate all the interactions inside the CCD. We've shown that CREME96 can be used. Also a cosine law distribution around the CCD is a very good approximation of real incident particles distribution (distribution confirmed by the Geant4 simulation analysis)

Concerning the future of the simulator
(personal point of view only)

Two options :

- **TARS implementation using Geant4 library (C++)**

- **Investigate the Geant4 vs. TARS error rate and implement this error rate in TARS** to conduct simple simulations and tests on CCD during the early development phase (without 3D model)

  As discussed wit Giovanni and Petteri, this would resolve the problem we have now for TARS but we will have to come back to Geant4 if we find other anomalies. We will also have to come back to Geant4 if we want to implement other tools for TARS (damages on Si induced by GCR interaction - > CTI simulations tools include in TARS).

  Developing a proper interaction simulation may be a waste of time considering that Geant4 is powerful enough and as been developed during more that 10 years now to be runnable on a classic computer really easily.

  Actually Geant4 is only a library that could be used for personalised tools (that the case of GRAS, SPENVIS, CIRSUS… etc.)

The first solution provides a strong base for the future CCD simulator implementation