Fe55 MC-data comparison and code improvements

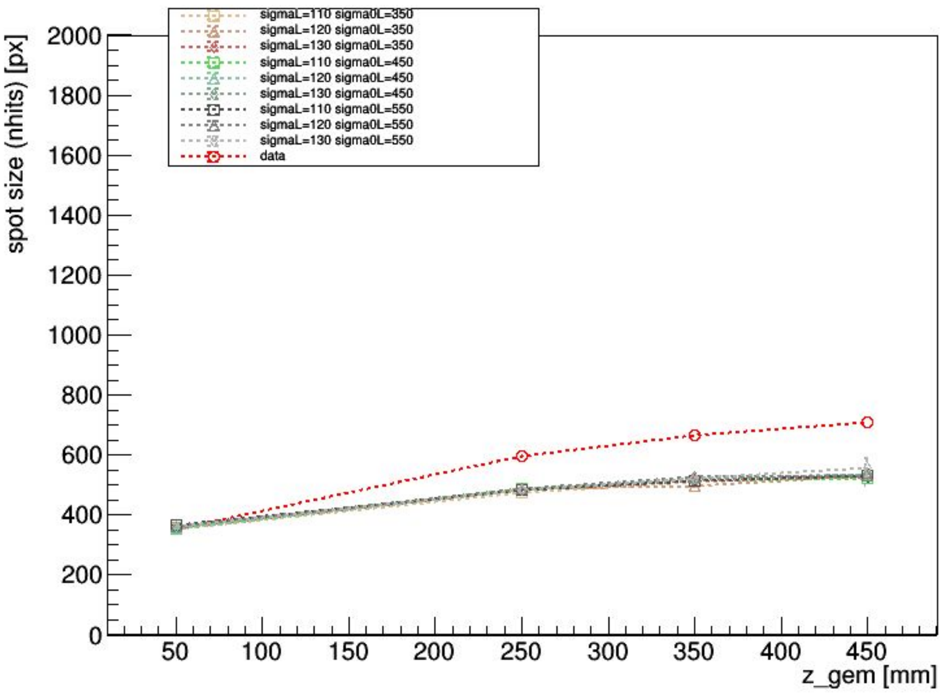
F. Petrucci, P. Meloni
16/05/2022

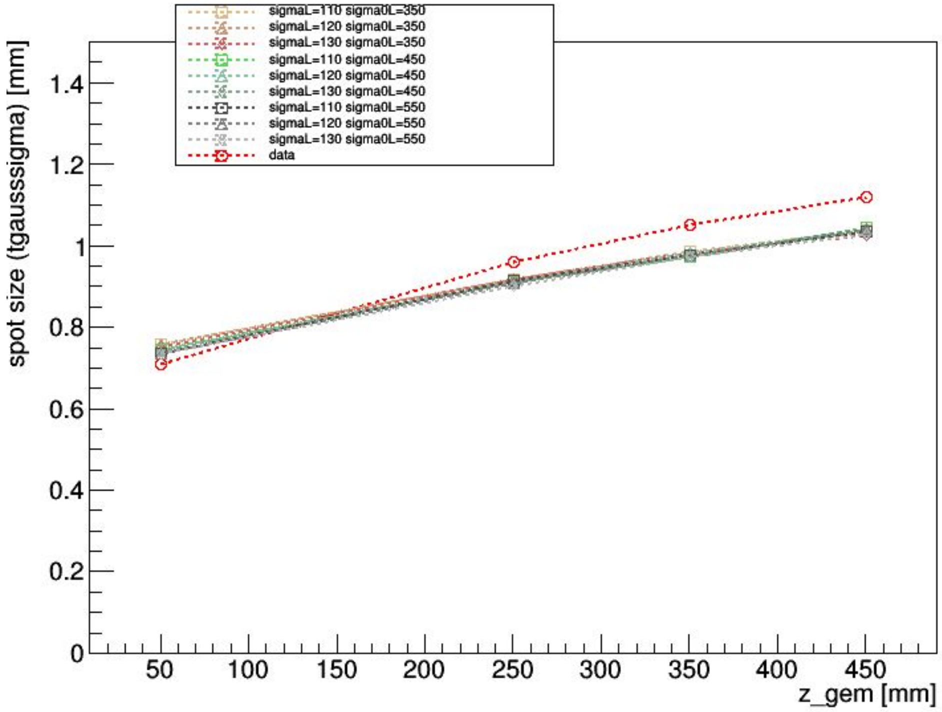
Overview

- scans for **sigmaL**, **absorbtion length, z_vox_dim, beta (again)**
- optimization code 1 (**image rebinning**) and **scan for x,y voxel dimension**
- optimization code 2 (**vectorized smearing**) to speed up the digitization

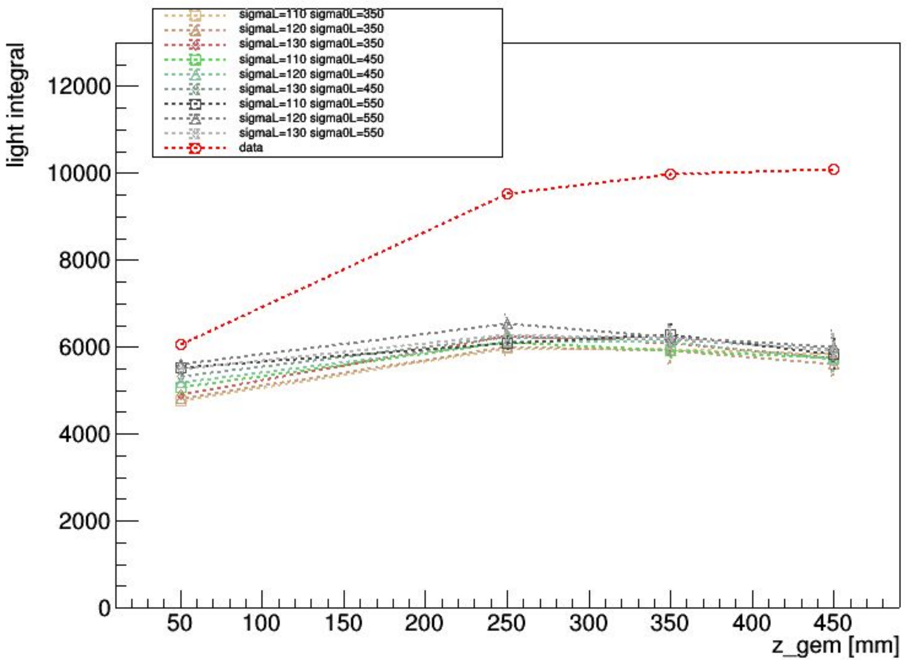
Varying **sigmaL & sigma0L**
with **sigmaT=130μm/√(cm)**
sigma0T = 550 μm
(A=1)

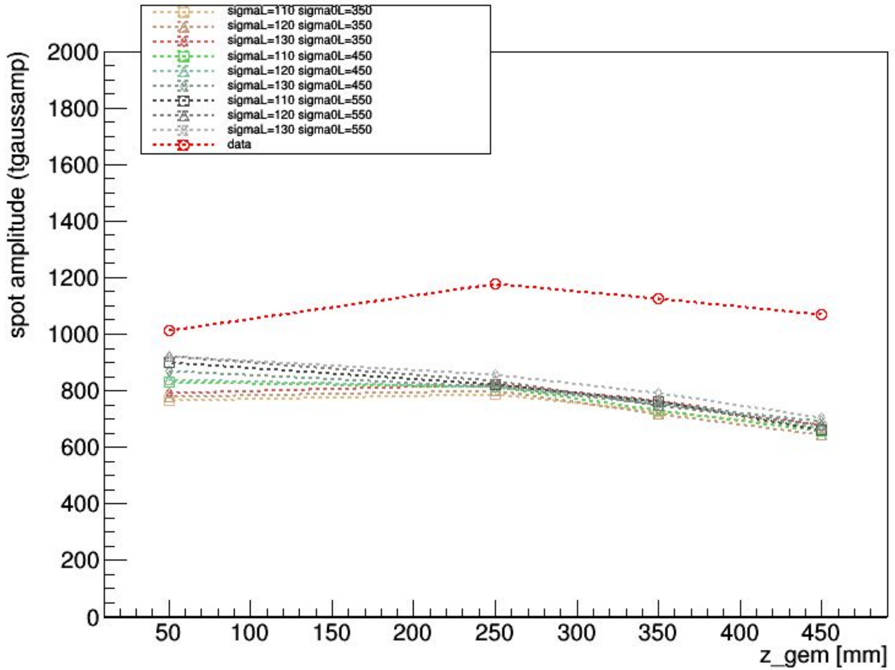
sigmaL and sigma0L scan (with sigmaT=130μm/√(cm) sigma0T = 550 μm, A=1)





Practically unchanged: as expected, longitudinal diffusion does not affect the spot size sigmaL and sigma0L scan (with sigmaT=130μm/√(cm) sigma0T = 550 μm, A=1)





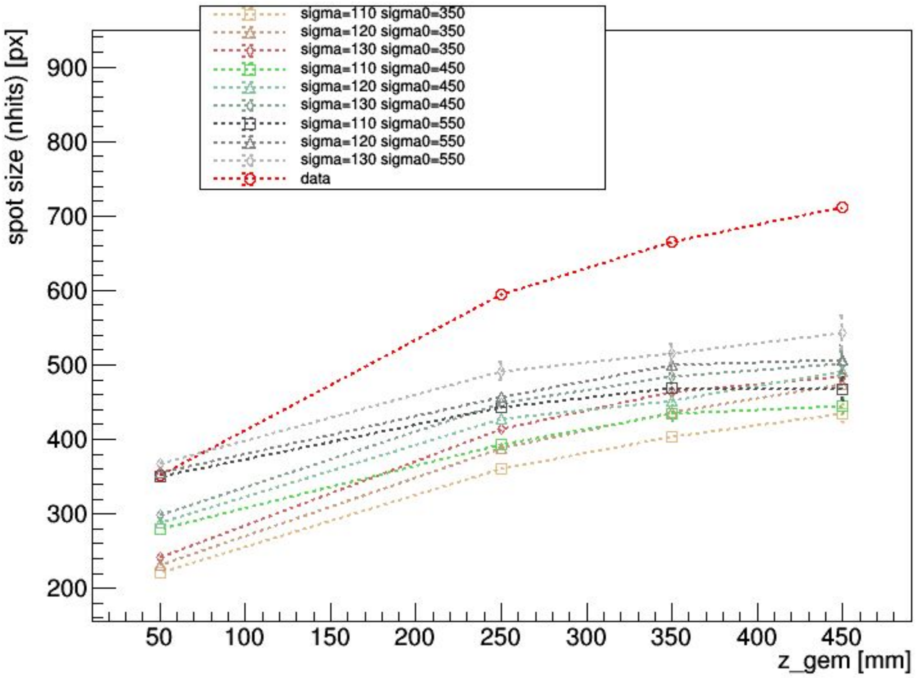
With high L. diffusion parameters, the trend at low z is ruined (due to the saturation): it’s better to keep sigmaL and sigma0L small. Original values were: **sigmaL = 99 μm/√(cm)** and **sigma0L = 260 μm** (not in the scan)

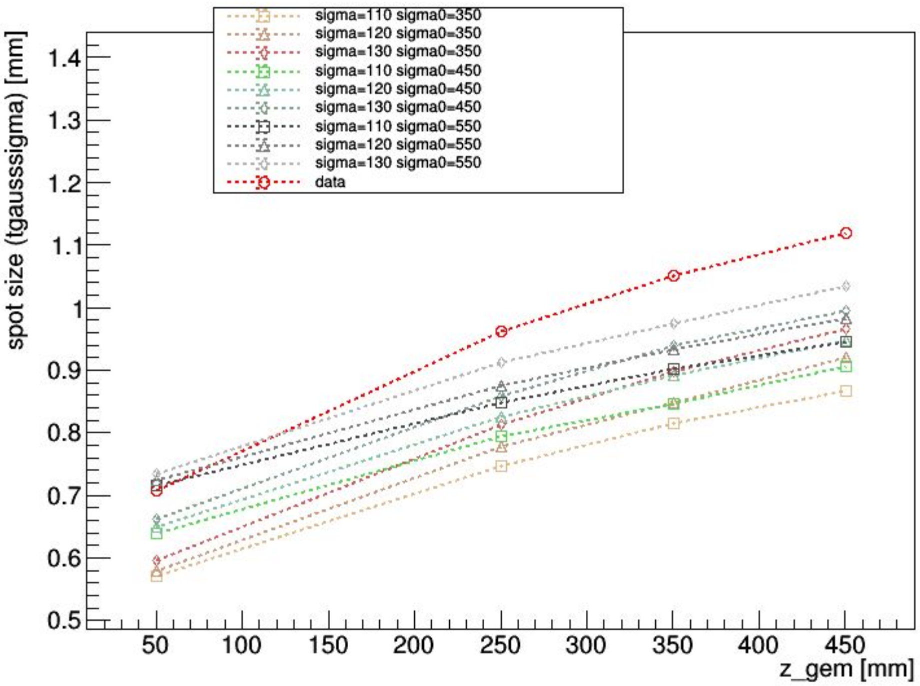
Varying all sigma together:

sigmaT, sigma0T, sigmaL, sigma0L

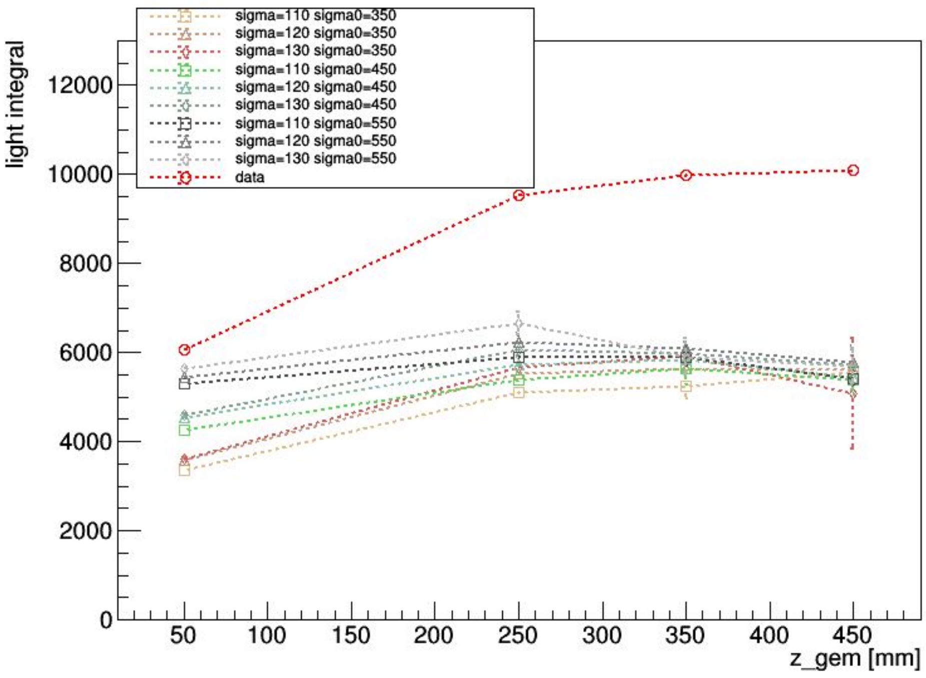
(A=1)

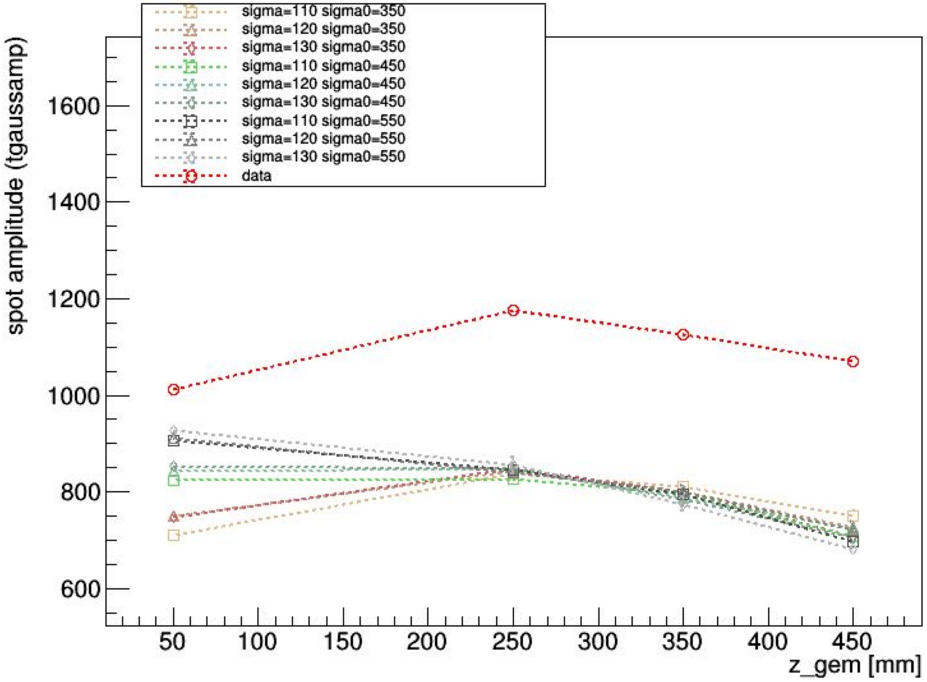
(A=1)


sigma=110 sigma0=350
sigma=120 sigma0=350
sigma=130 sigma0=350
sigma=110 sigma0=450
sigma=120 sigma0=450
sigma=130 sigma0=450
sigma=110 sigma0=550
sigma=120 sigma0=550
sigma=130 sigma0=550
data
spot size (nhits) [px]
z_gem [mm]


sigma=110 sigma0=350
sigma=120 sigma0=350
sigma=130 sigma0=350
sigma=110 sigma0=450
sigma=120 sigma0=450
sigma=130 sigma0=450
sigma=110 sigma0=550
sigma=120 sigma0=550
sigma=130 sigma0=550
data
spot size (tgaussigma) [mm]
z_gem [mm]

(A=1)





With high L. diffusion parameters, the trend at low z is ruined (due to the saturation): it’s better to keep sigmaL and sigma0L small (yellow line)

Try by varying abs_lenght

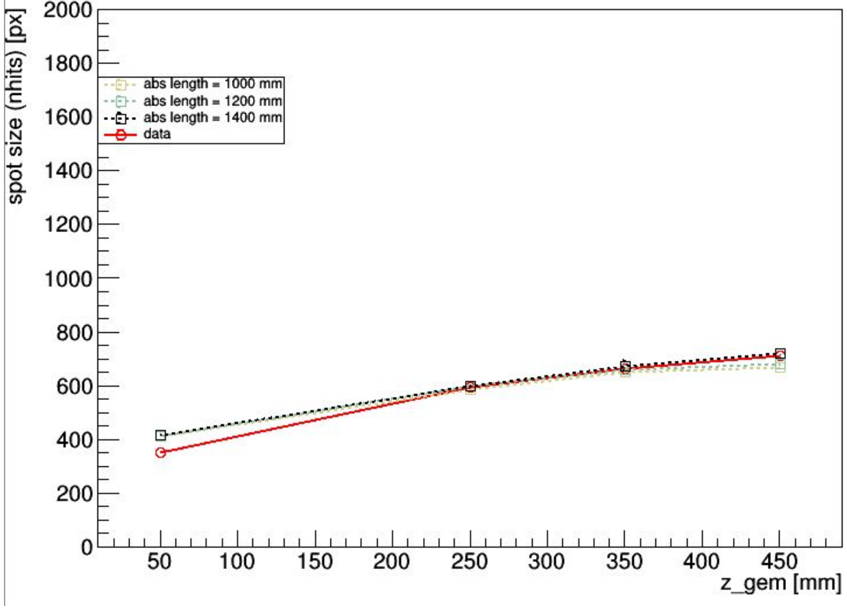
with sigmaT= 130μm/√(cm)

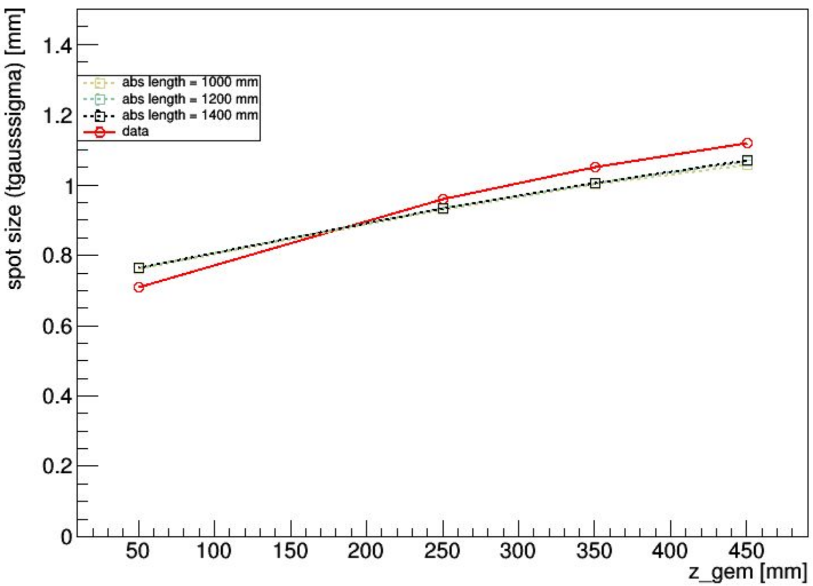
sigma0T = 550 μm

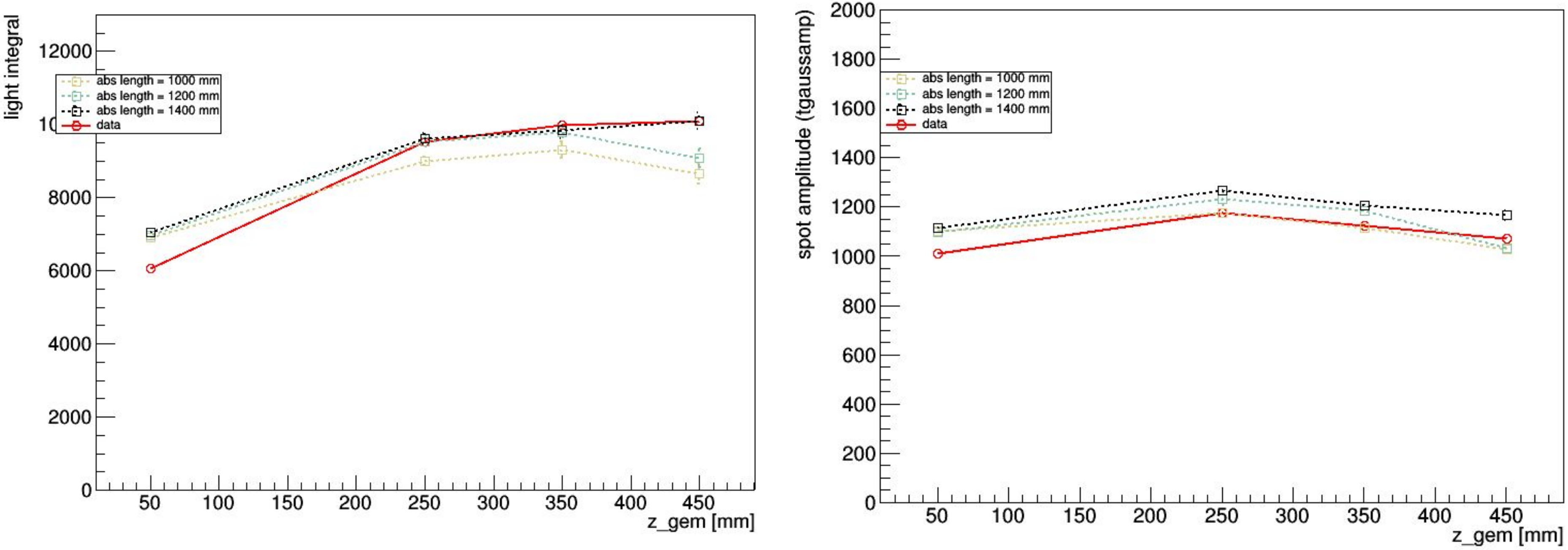
A=1.6

sigmaL = 99 μm/√(cm)

sigma0L = 260 μm


spot size (nhits) [px]
abs length = 1000 mm
abs length = 1200 mm
abs length = 1400 mm
data
z_gem [mm]


spot size (tgaussigma) [mm]
abs length = 1000 mm
abs length = 1200 mm
abs length = 1400 mm
data
z_gem [mm]



1400 mm looks better for the integral, but not so good for the spot amplitude (unless A is slightly decreased)

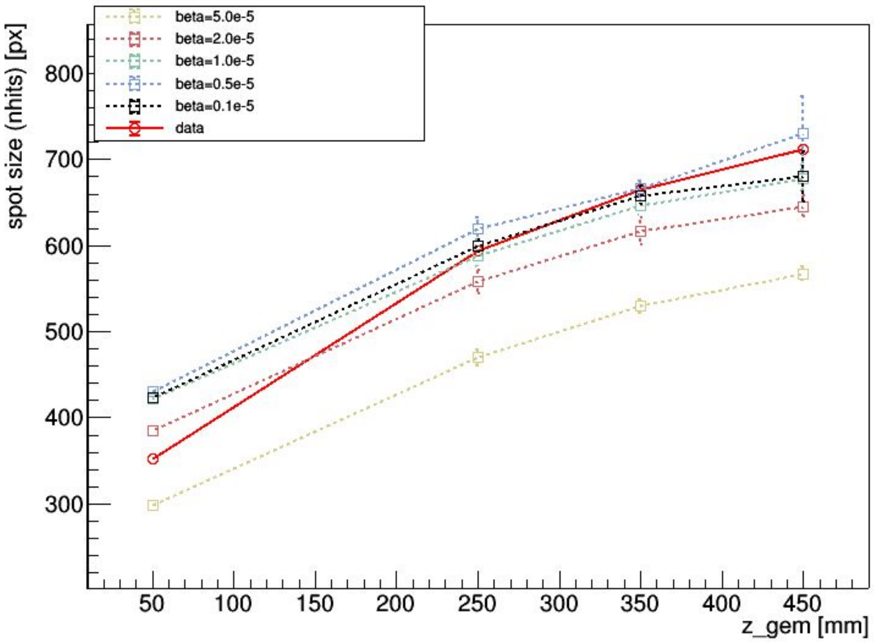
Trying by varying “beta”

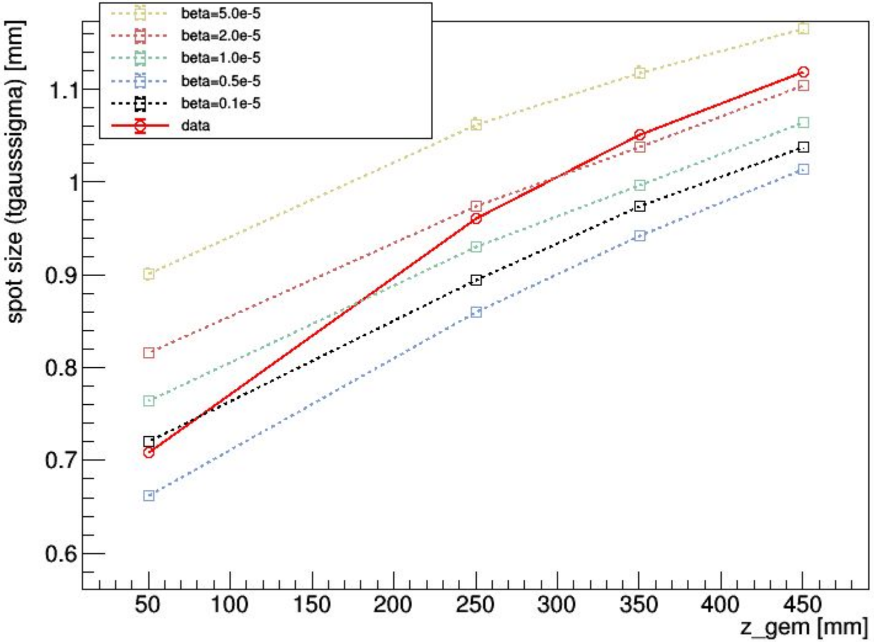
with sigmaT=130µm/√(cm)

sigma0T = 550 µm

<u>A=1.6</u>

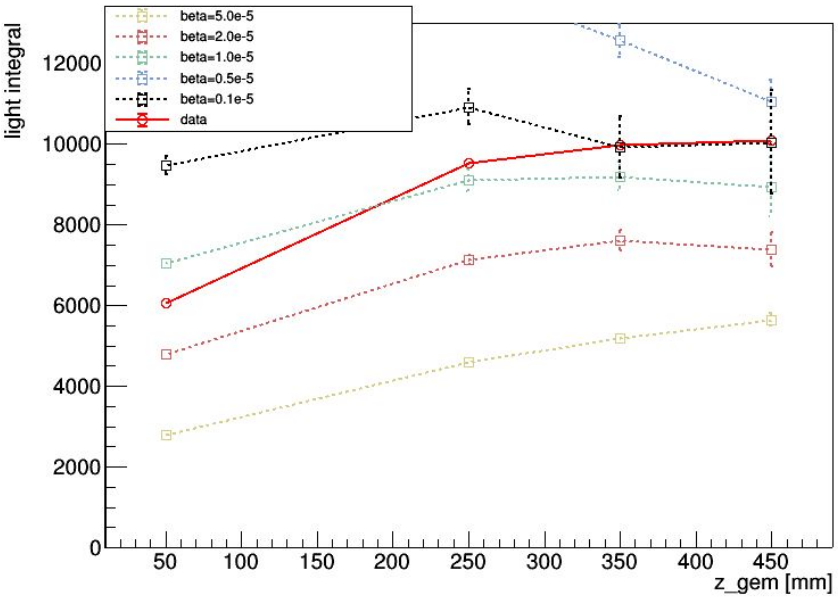
z scan (different beta)

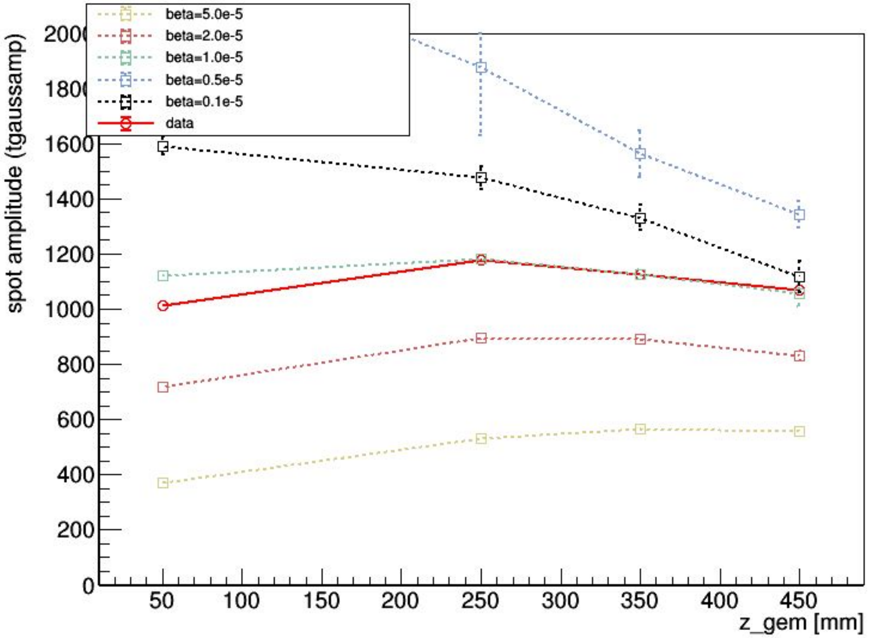




Best parameter: beta=1.0e-5 (original value) -> **is the best value since number of hits decreases with beta, and tgauss_sigma increases with beta**

z scan (different beta)





Best parameter: beta=1.0e-5 (standard value)

Try by varying z_vox_dim

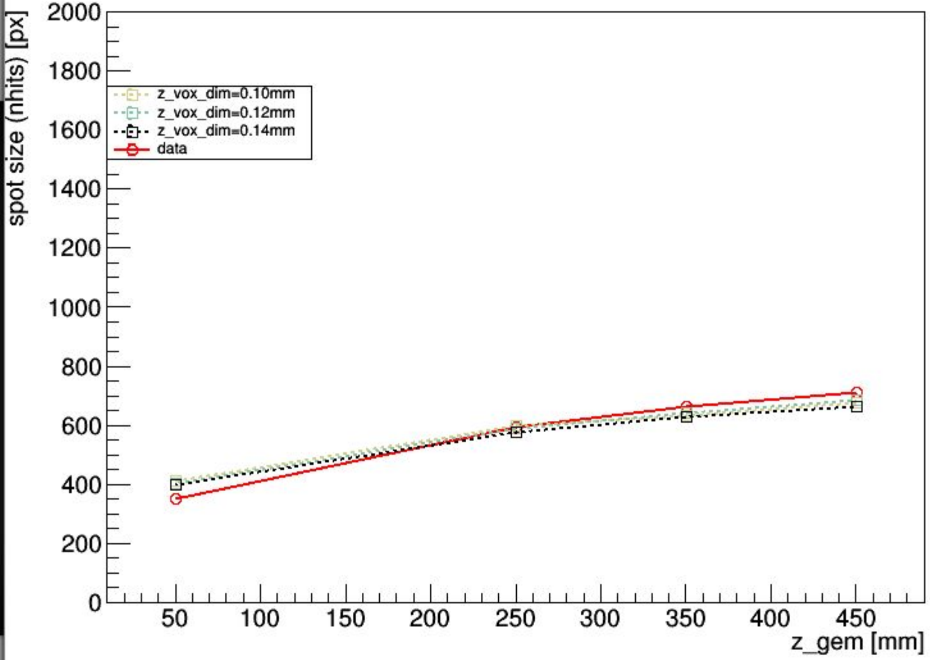
with sigmaT= 130μm/√(cm)

sigma0T = 550 μm

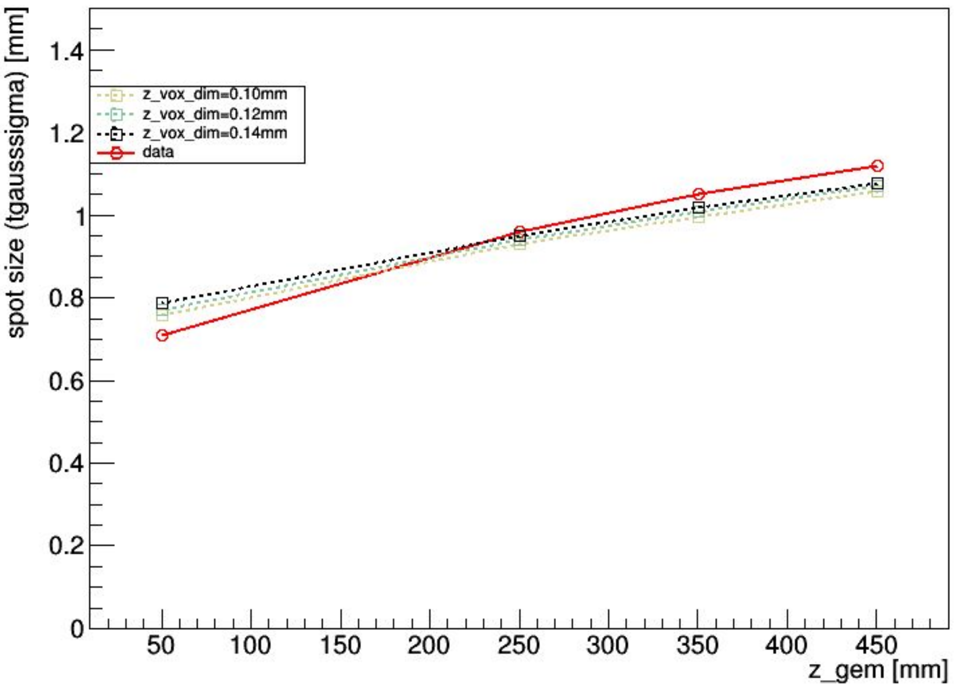
A=1.6

sigmaL = 99 μm/√(cm)

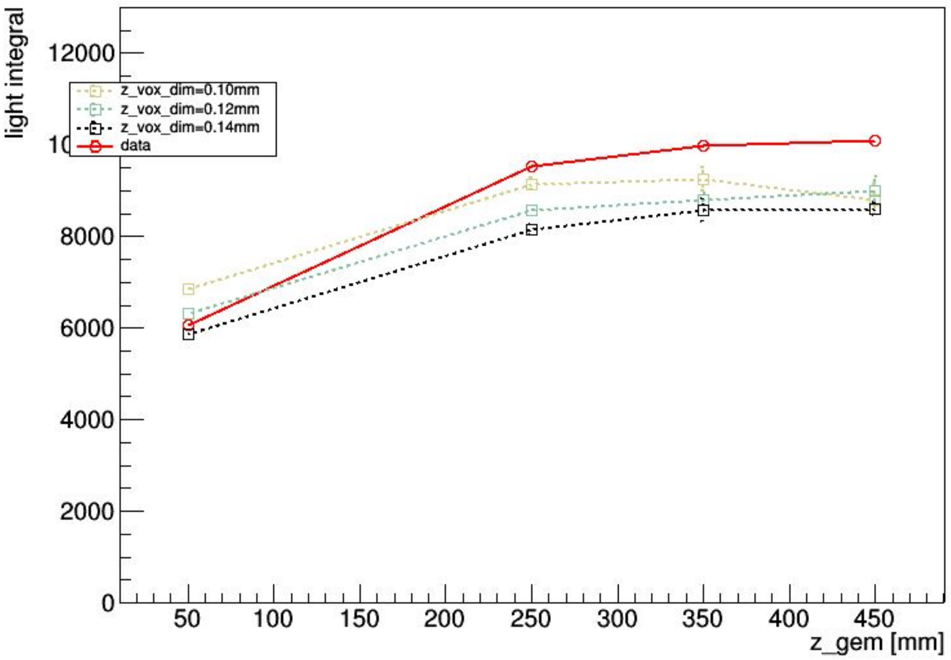
sigma0L = 260 μm



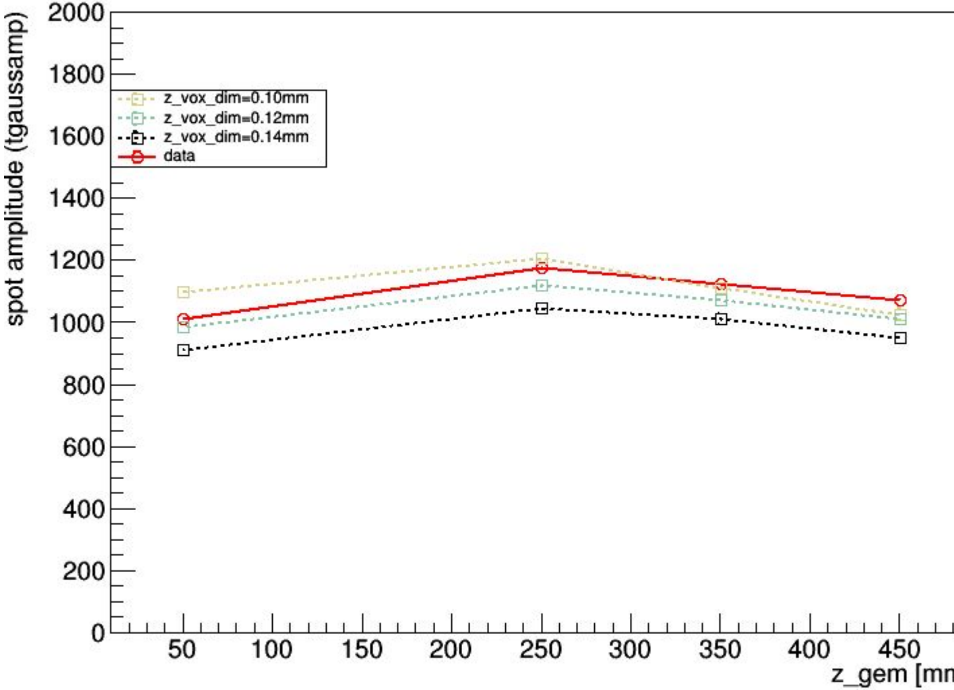
spot size (nhits) [px]
2000
1800
1600
1400
1200
1000
800
600
400
200
0
50
100
150
200
250
300
350
400
450
z_gem [mm]
z_vox_dim=0.10mm
z_vox_dim=0.12mm
z_vox_dim=0.14mm
data



spot size (tgaussigma) [mm]
1.4
1.2
1
0.8
0.6
0.4
0.2
0
50
100
150
200
250
300
350
400
450
z_gem [mm]
z_vox_dim=0.10mm
z_vox_dim=0.12mm
z_vox_dim=0.14mm
data



light integral
z_vox_dim=0.10mm
z_vox_dim=0.12mm
z_vox_dim=0.14mm
data
0
2000
4000
6000
8000
12000
50
100
150
200
250
300
350
400
450
z_gem [mm]



spot amplitude (tgaussamp)
z_vox_dim=0.10mm
z_vox_dim=0.12mm
z_vox_dim=0.14mm
data
0
200
400
600
800
1000
1200
1400
1600
1800
2000
50
100
150
200
250
300
350
400
450

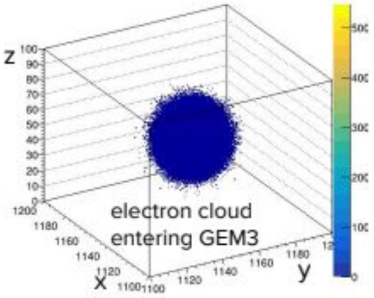
Optimization code 1: rebinning problem

adding possibility to change x,y voxel dimensions (so far we used the “pixel dimensions”)

Problem: in the current digitization code there is no way to set x, y voxel dimensions. They are fixed to be = **detector_dimension / number_of_pixels.** (“pixel dimension”)

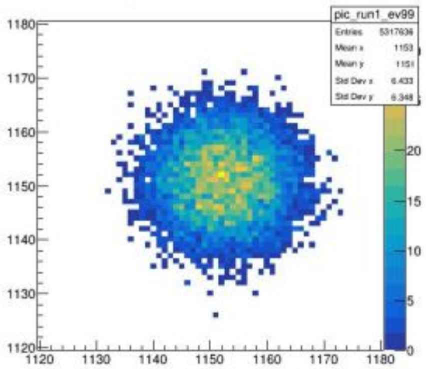
Once the saturation is applied, you project the 3D histo on the x and y axes, getting a 2D histo that has 2304x2304 bins. Once you scale that histo by the photons_per_electron factor and the solid angle, you get the final image.

You can change the x, y voxel dimensions only by changing detector or camera (n. pixels).



3D histogram to apply saturation effect

simply projecting on x and y plane →

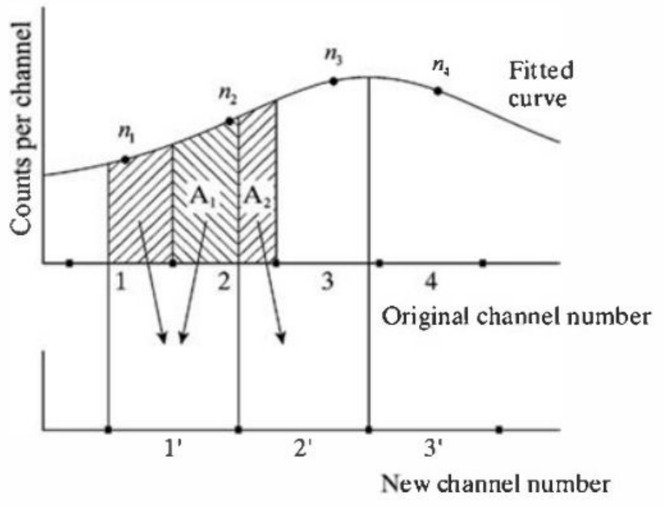


Final image after saturation and scaling by optical factors

Solution: a library that uses a solution explained by G. Knoll
https://github.com/jhykes/rebin

1. Fit to the histogram (with a spline function).
2. Compute the area under the fitted curve between the edges of the nearest new bins/channels.
3. Compute the fractions of this area that overlap the original bin edges (A_1 and A_2)
4. Use these fractions as weighting factors in reassigning the counts in an original bin to one or more new bins (depending on the degree of overlap).

$A_1/(A_1+A_2)$ for 1' and $A_2 / (A_1+A_2)$ for 2'

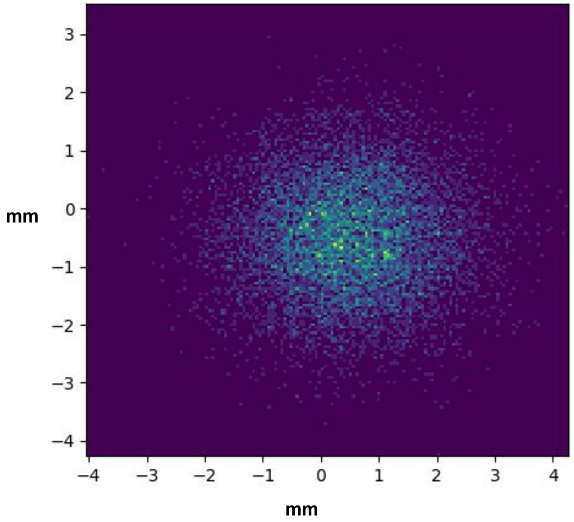


[Glenn Knoll, Radiation Detection and Measurement, 3rd ed., Wiley, 2000.]

detector dim / n. pixel

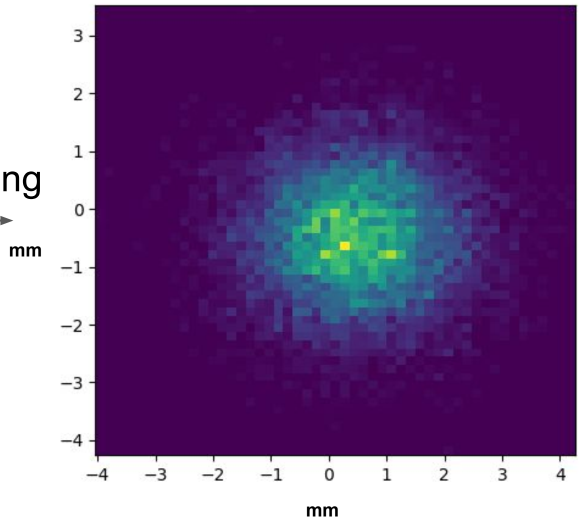
Case 1: x,y voxels dim [mm] < "pixel dim" [mm]

Example: x,y voxel dim = 0.05 mm
pixel dim = 0.15 mm (correct value in LIME)



1 bin = 1 voxel

rebinning

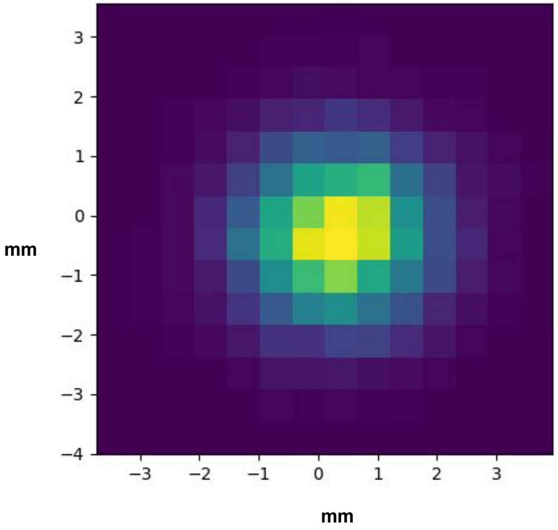


1 bin = 1 pixel

detector dim / n. pixel

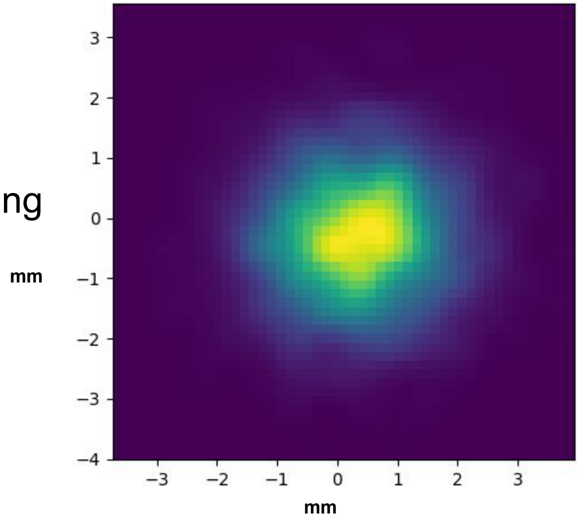
Case 2: x,y voxels dim [mm] > “pixel dim” [mm]

Example: x,y voxel dim = 0.5 mm
pixel dim = 0.15 mm (correct value in LIME)



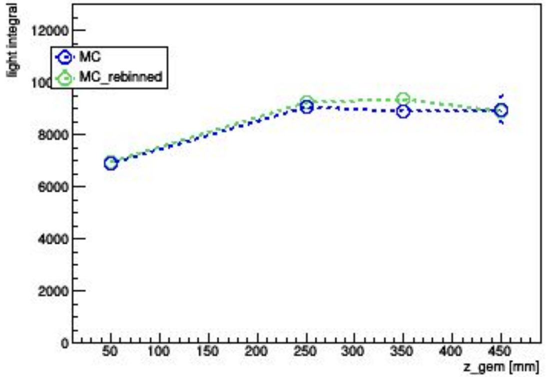
1 bin = 1 voxel

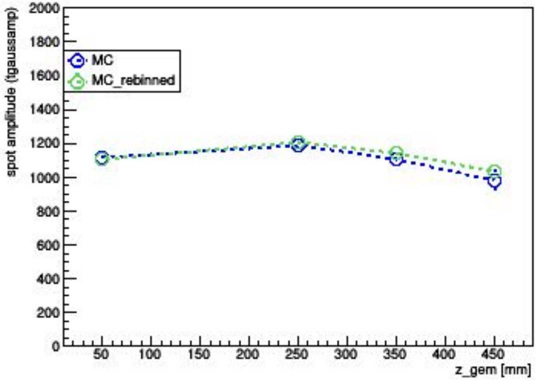
rebinning

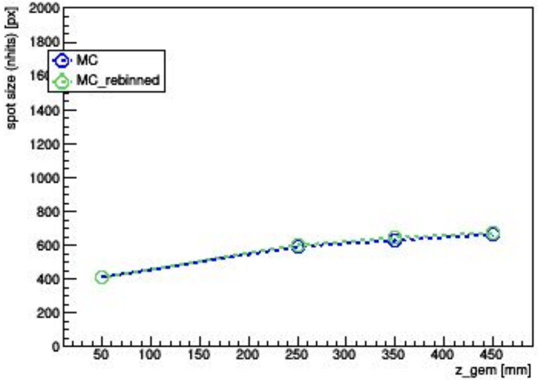


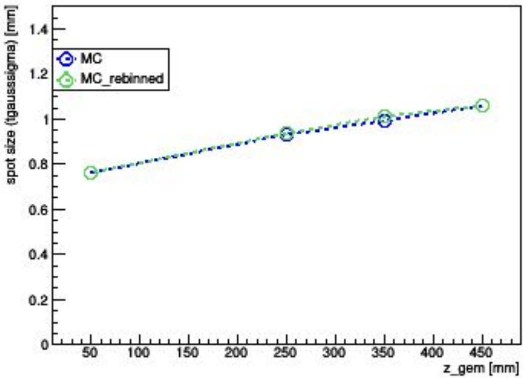
1 bin = 1 pixel

Comparing MC with vs without rebinning (z scan, same parameters)



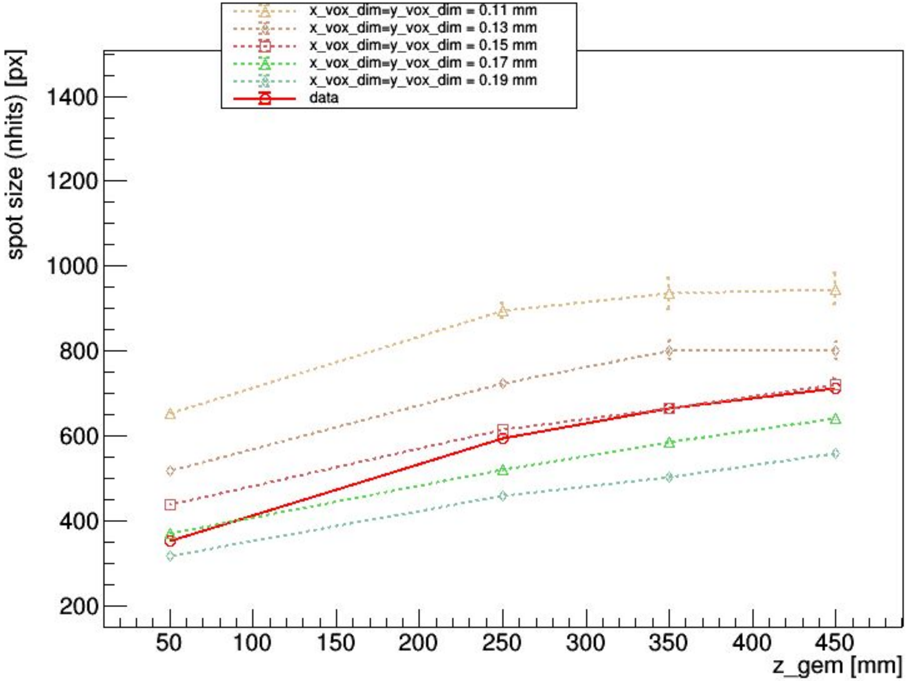


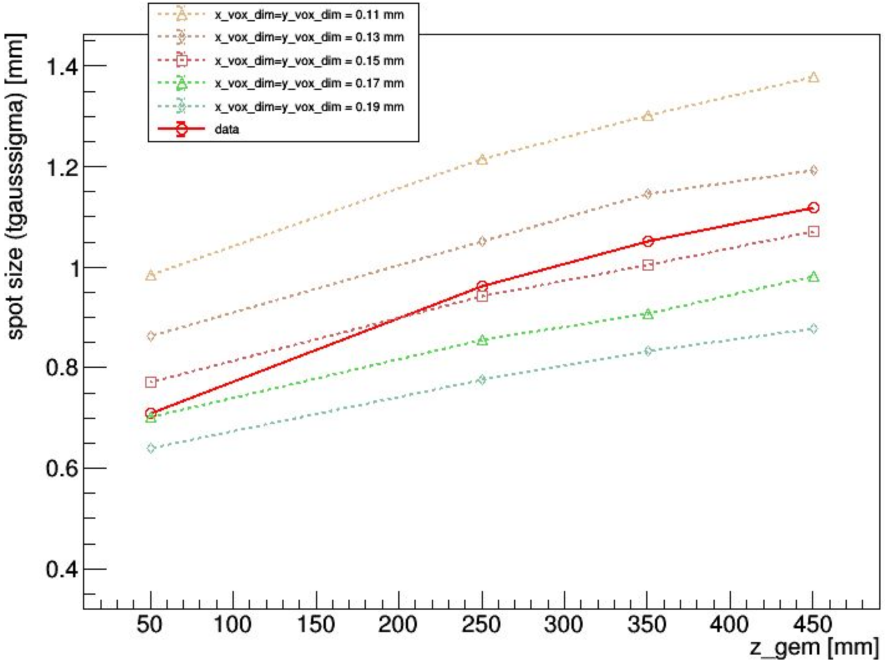




Setting the same parameters, the two codes seems to give the same output

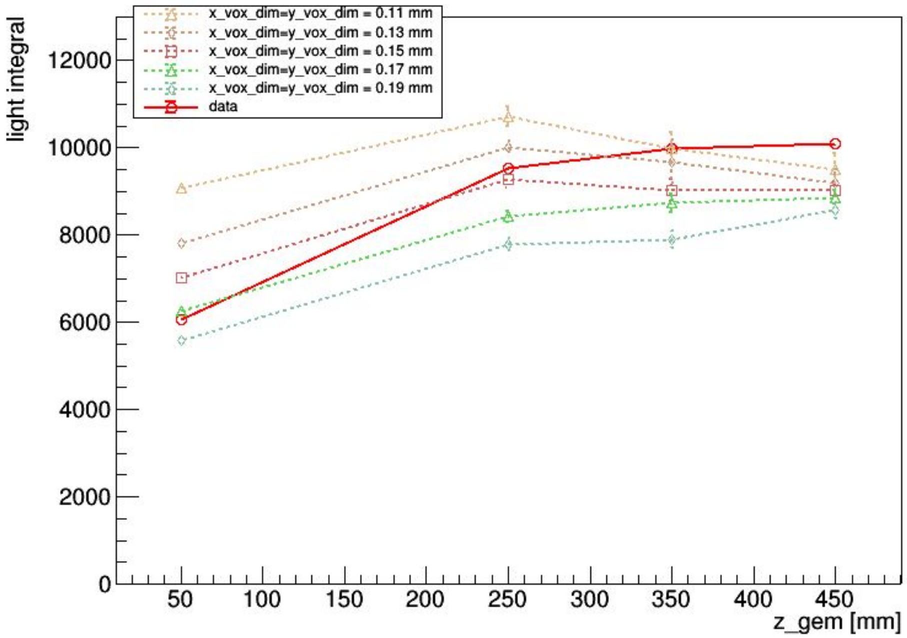
Scan with different values of x, y voxel dimension (A=1.6, max sigmaT, min simgaL)

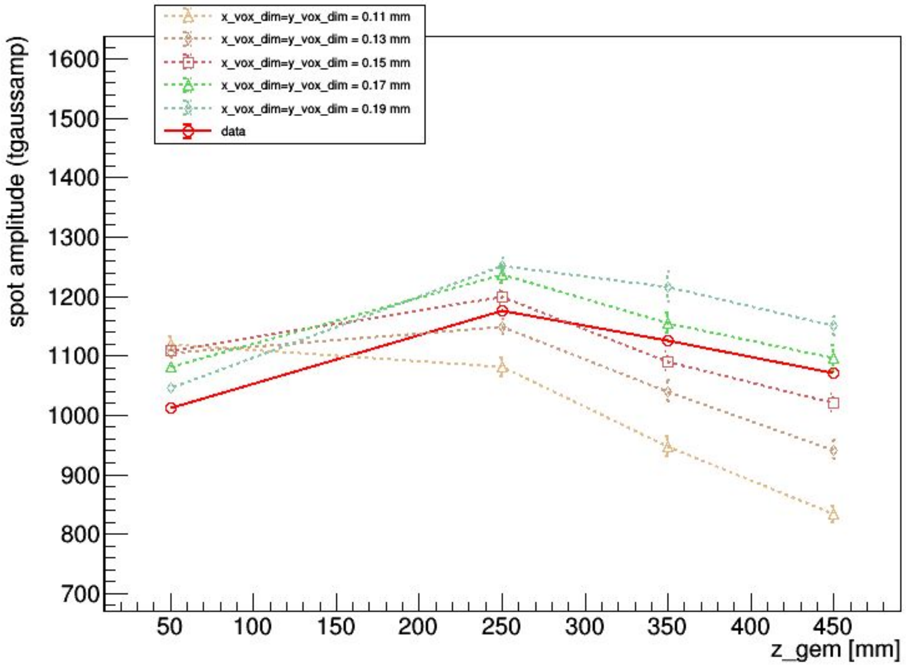




NOTE: the standard value for x,y vox dimension is: (detector_dim / n_pixels) = 346 mm /2304 ~0.15 mm

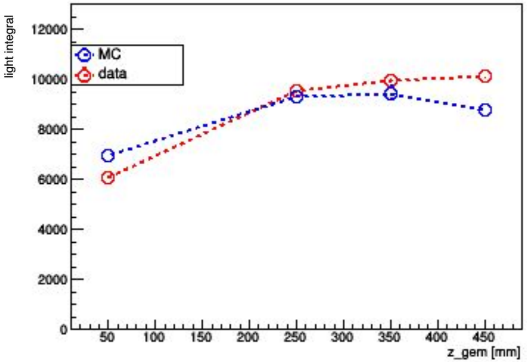
Scan with different values of x, y voxel dimension (A=1.6, max sigmaT, min simgaL)

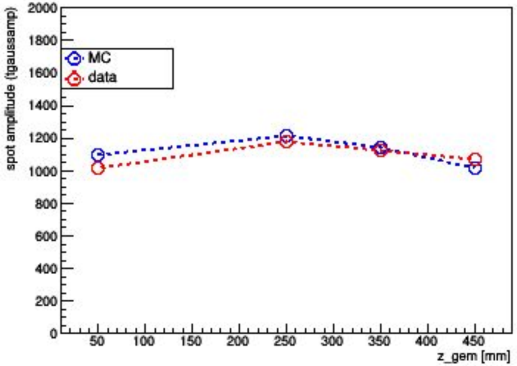


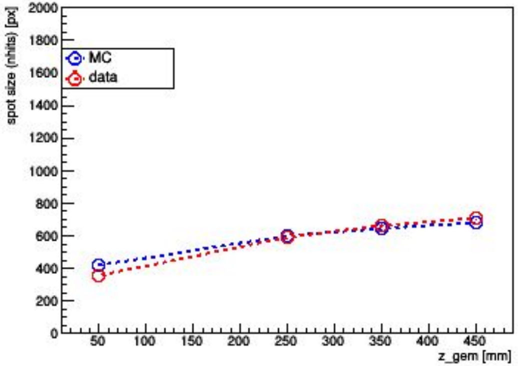


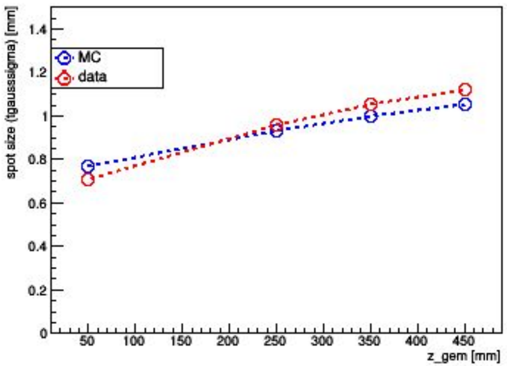
NOTE: the standard value for x,y vox dimension is: (detector_dim / n_pixels) = 346 mm /2304 ~**0.15 mm**

Best result MC/data comparison









Digitization parameters

events per run= 100 events
detector dimensions = 346cm x 346cm
pixels = 2304 x 2304
pedestal = run 4432

beta= 1e-5
A= 1.6

x_vox_dim=0.15 mm
y_vox_dim=0.15 mm
z_vox_dim=0.1 mm

abs_len= 1000mm
z_gem = 5, 25, 35, 45 cm
GEM1_HV= 440V
GEM2_HV= 440V
GEM3_HV= 440V

diff_const_sigma0T= 0.3025 mm^2 (550 μm)
diff_coeff_T= 0.0169 [mm/sqrt(cm)]^2 (130 μm/sqrt(cm

diff_const_sigma0L= 0.0676 mm^2 (260 μm)
diff_coeff_L= 0.00978 [mm/sqrt(cm)]^2 (99 μm/sqrt(cm))

ion_pot = 0.0462 keV
photons_per_el = 0.07
counts_per_photon = 2.,

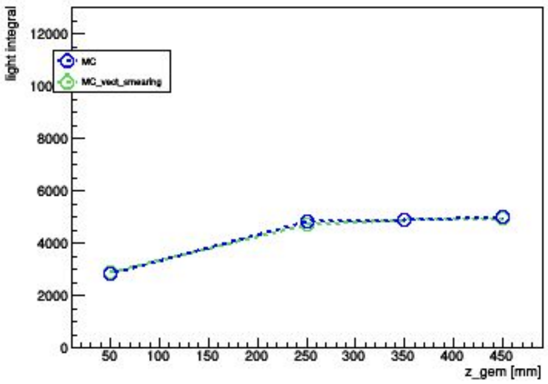
Optimization code2: speeding up the code

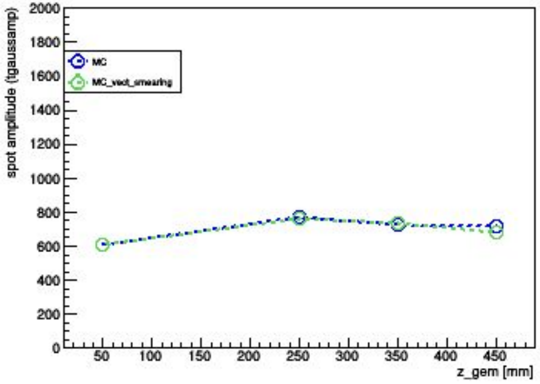
Expecting a slower code with this new “rebinning feature”

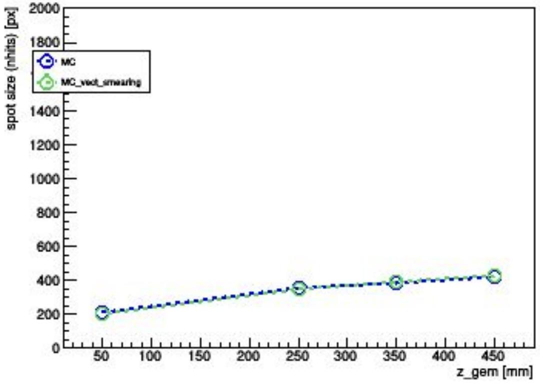
Improved by **vectorizing the smearing with numpy***: no for loop over hits (this helps when there are lots of hit).

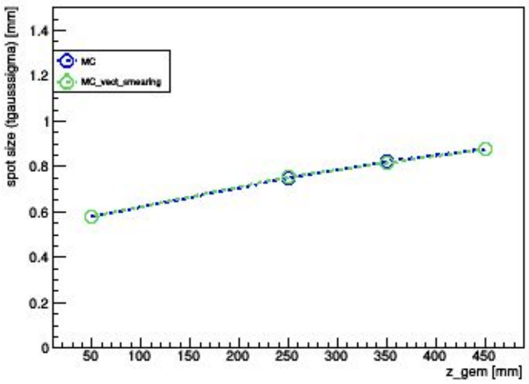
*A similar improvement was already done in December 2021 for the saturation effect only

Comparing MC with vs without “vectorized smearing”



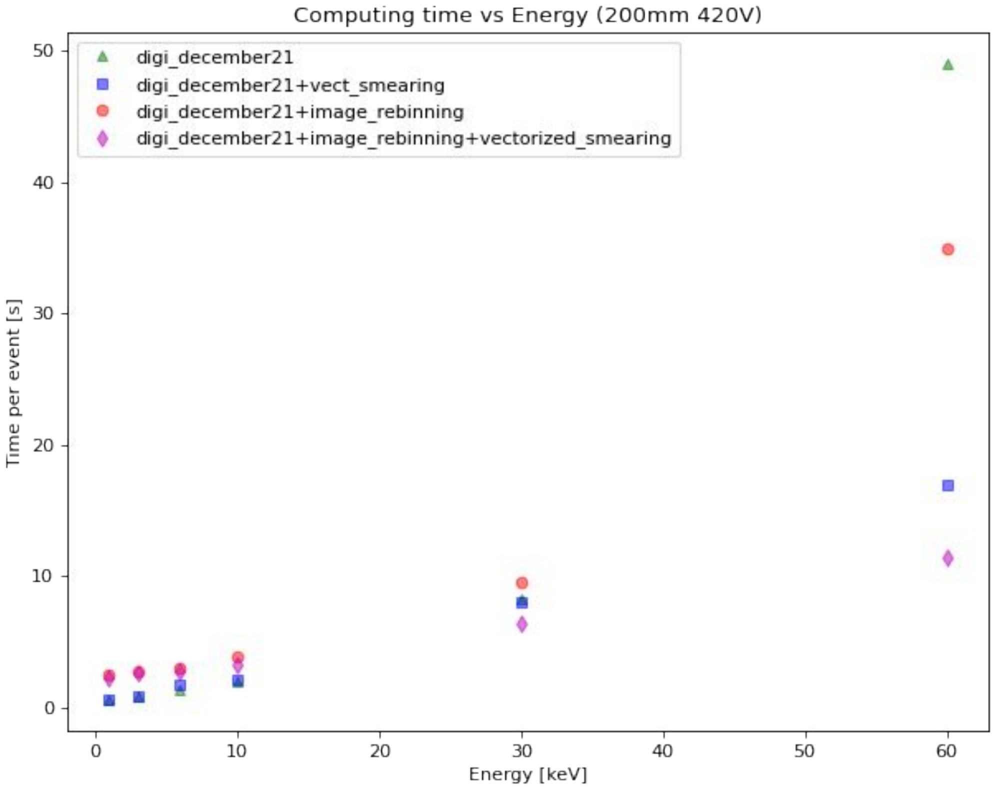






Setting the same parameters, the two codes seems to give the same output.

Computing time vs track energy



Not clear why the “image rebinning” code is faster than “digi_december21” at 60 keV. Need to look at the reconstructed tracks at 60 keV