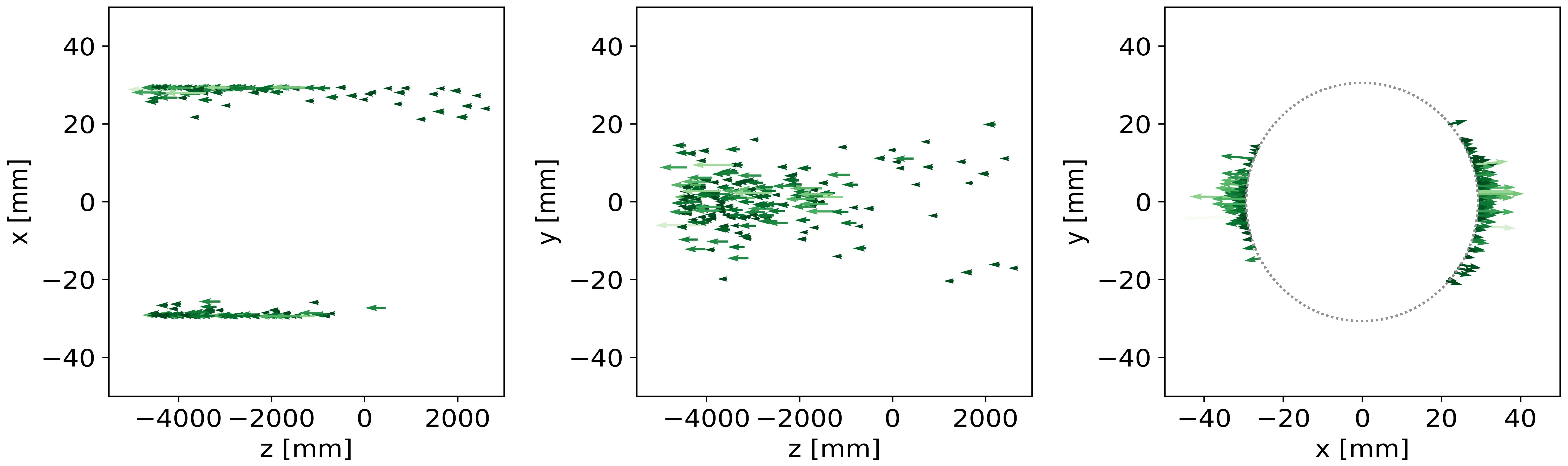
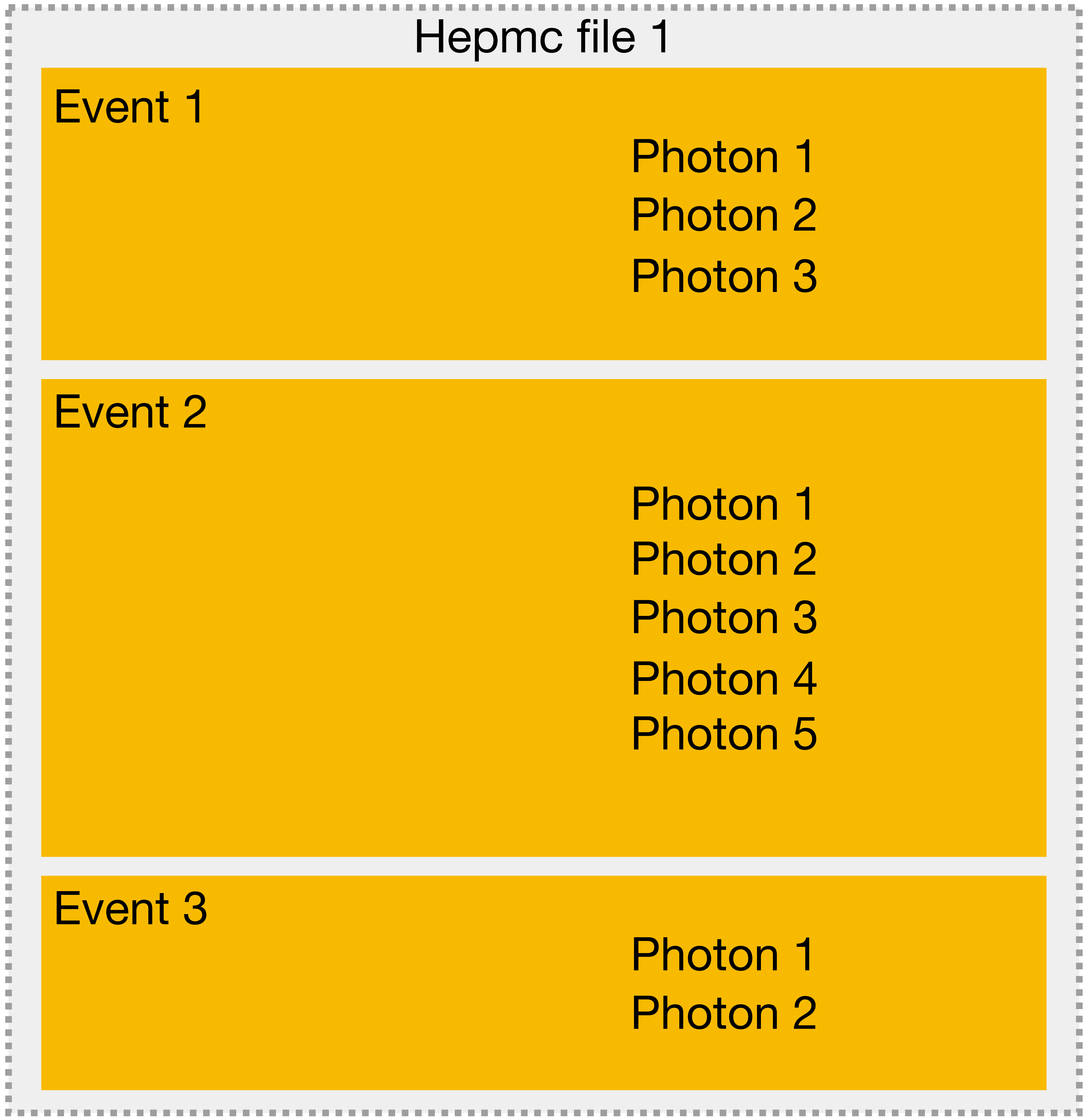
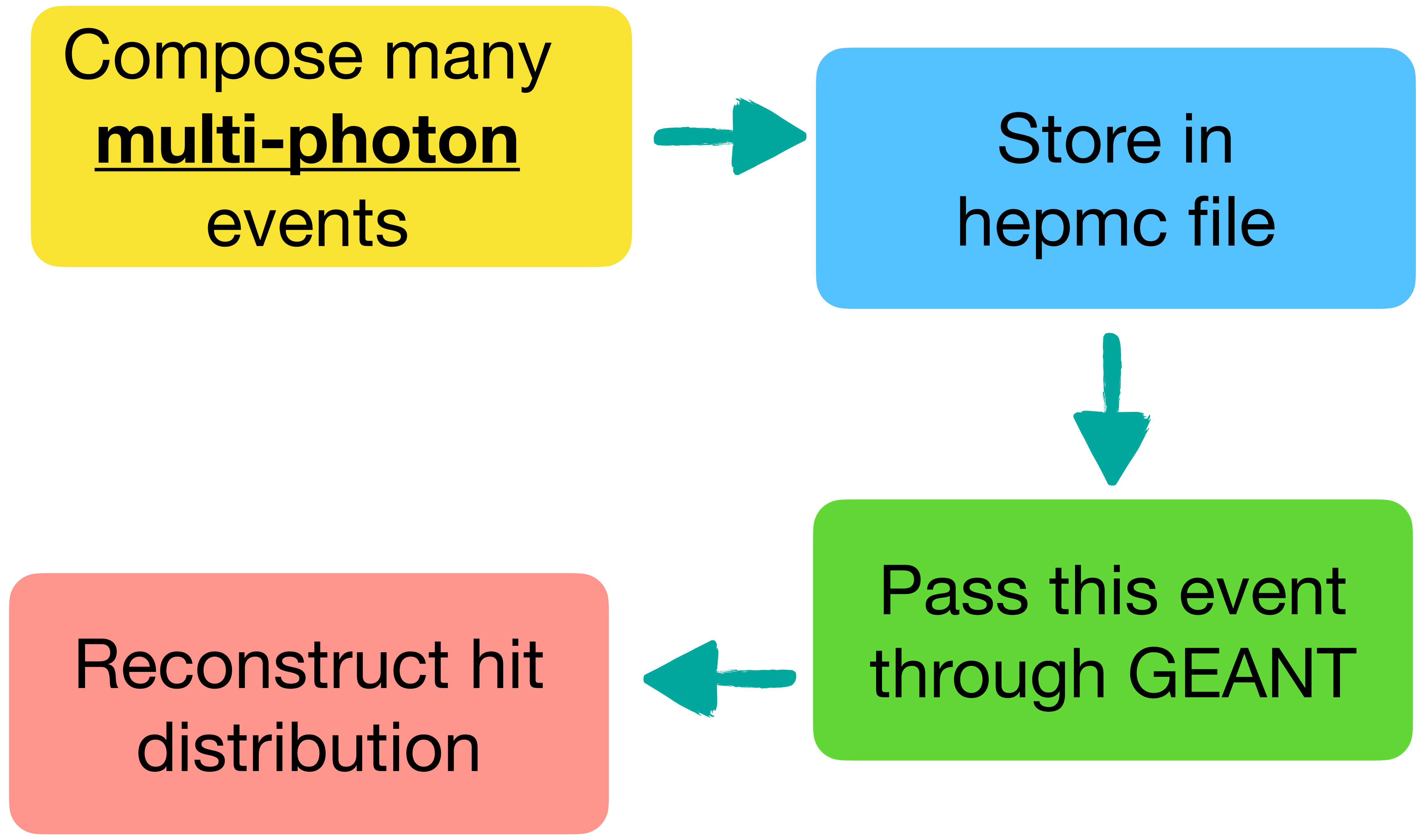
Updates on SR background



Rey Cruz-Torres
EPIC Background Meeting
12/16/2022

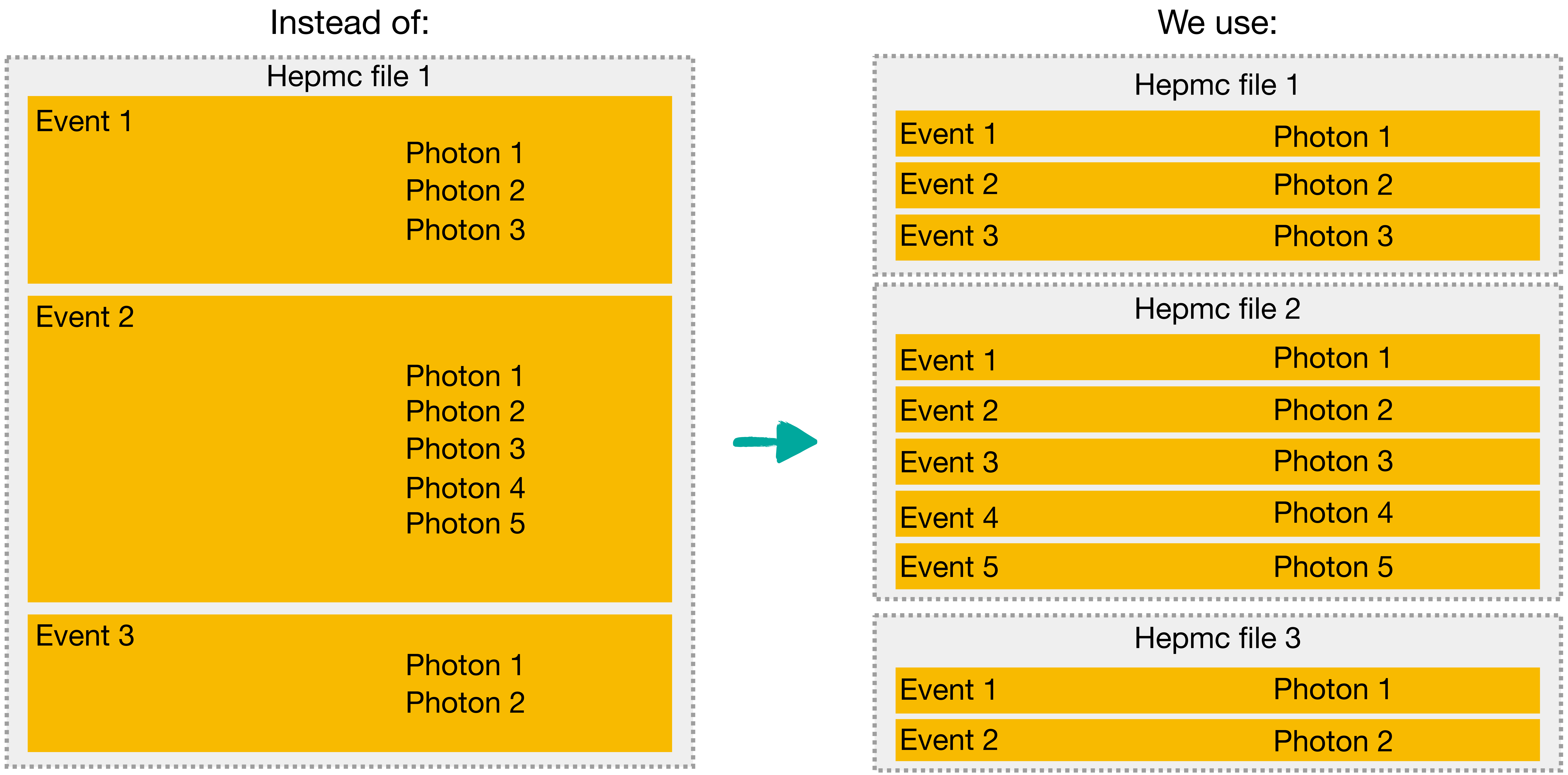
To recap




Compose many **multi-photon** events
Store in hepmc file
Pass this event through GEANT
Reconstruct hit distribution

Issues:
- DD4HEP hit distributions revealed that photon momentum vectors were detached from their respective vertices and launched from v = (0,0,0), which produces wrong topology

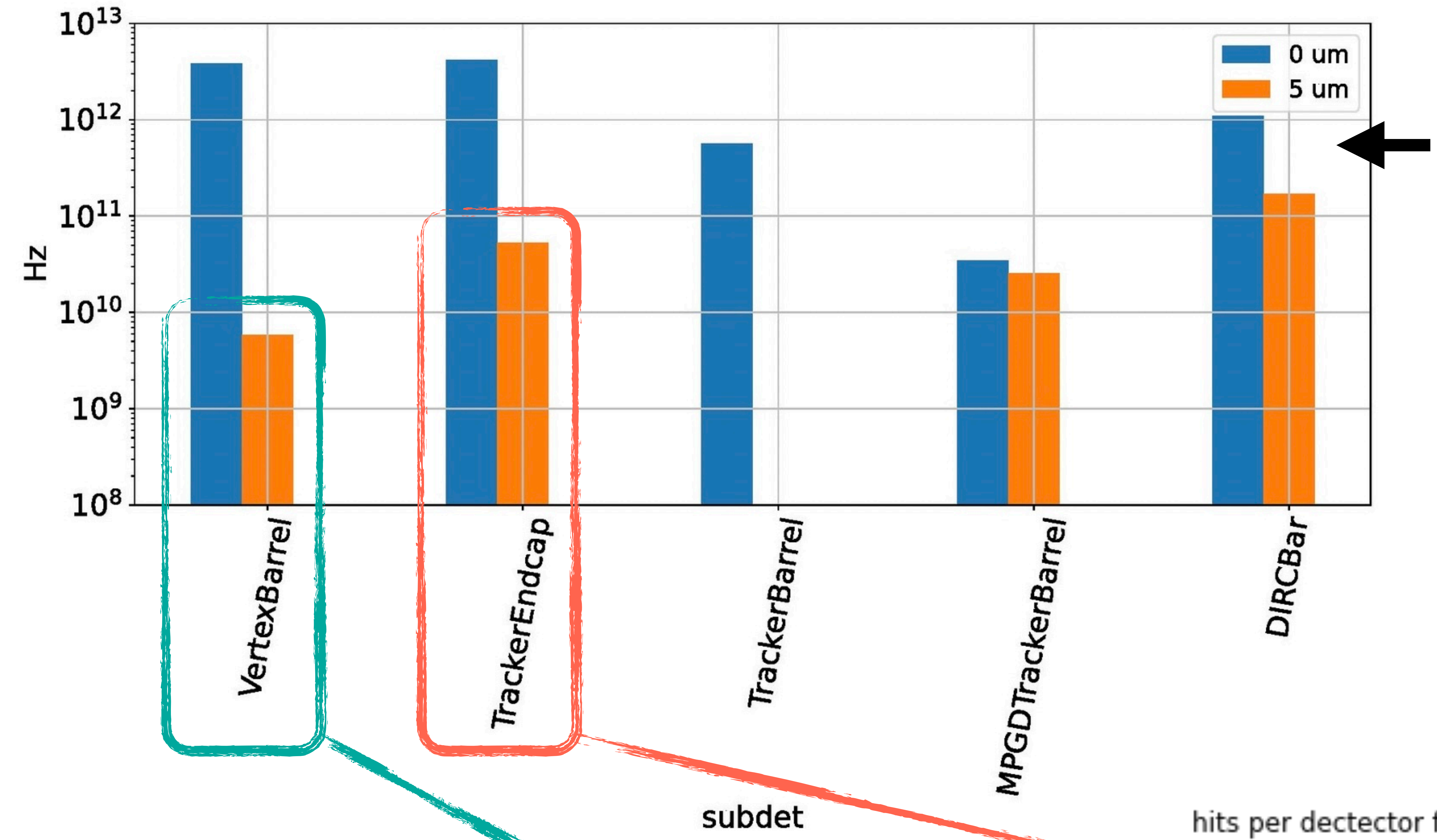
Workaround



* Implemented by UC Berkeley undergrad B. Sterwerf

Comparison to previous results

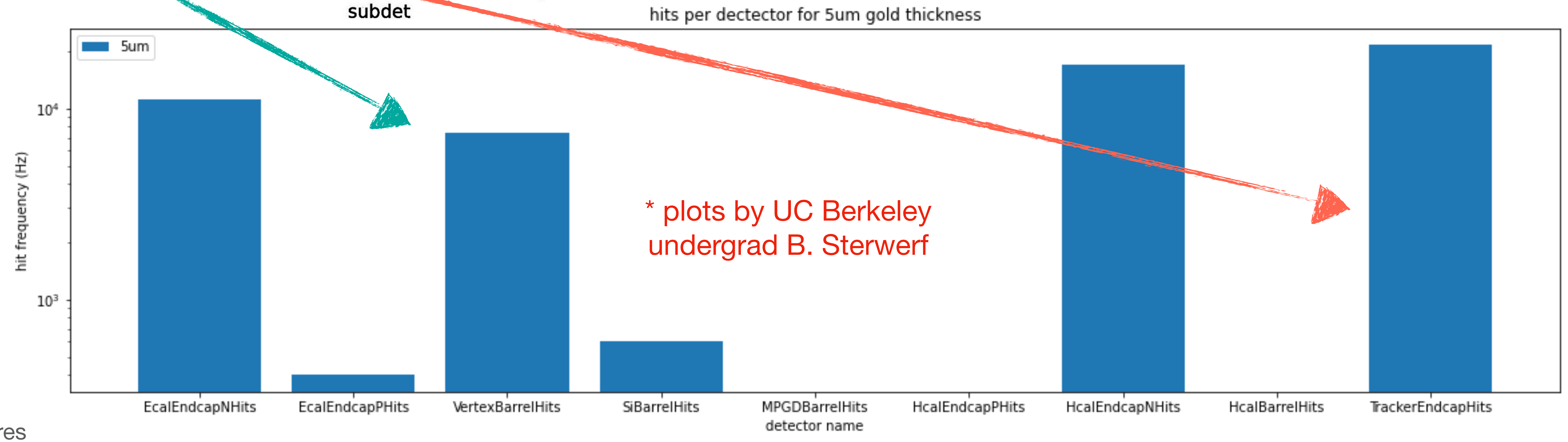
flux (for 2.5 A electron beam of E = 10 GeV)



Old (biased) method in which we passed all photons through Geant only once and scaled the resulting contribution by the provided weight.

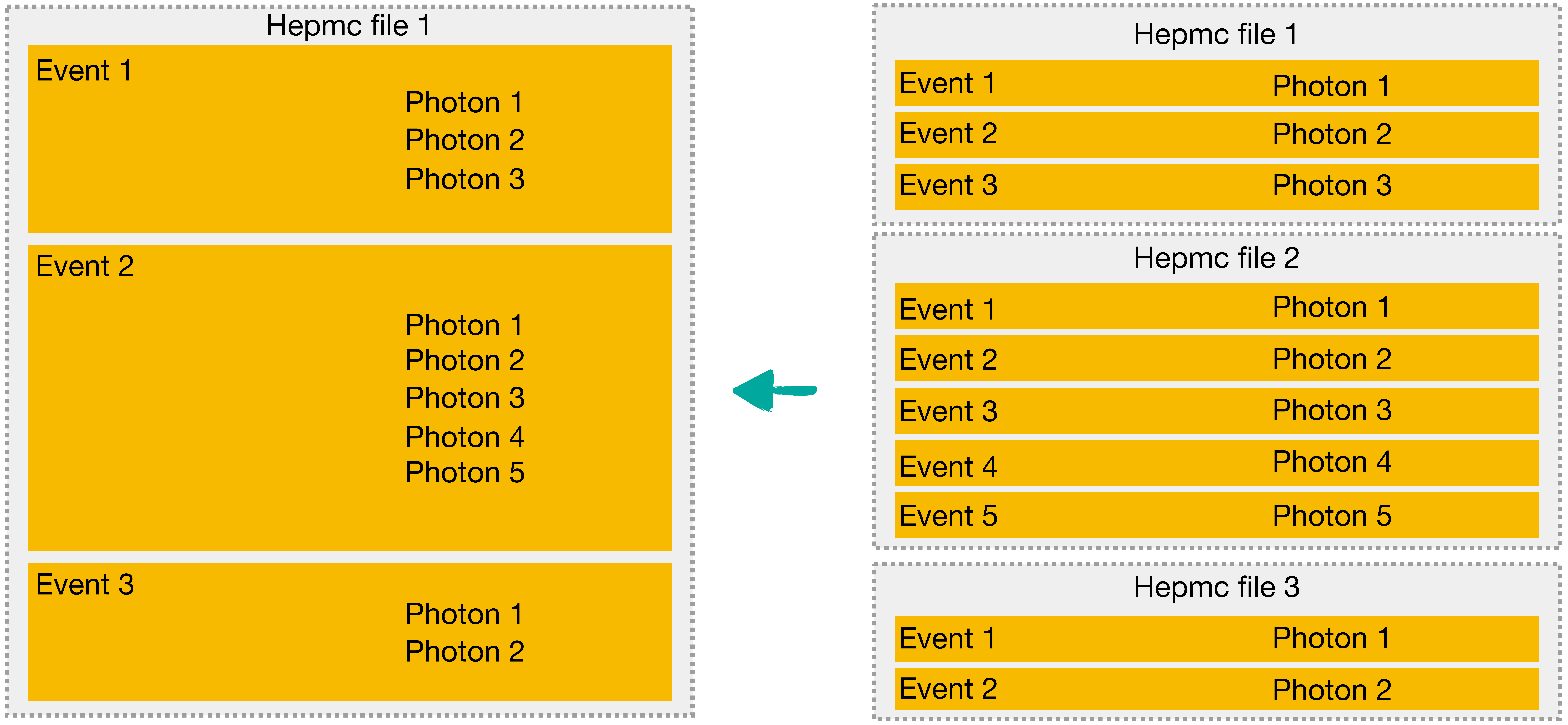
* Keep in mind the detector versions between these plots is different

Compare orange on top to blue on bottom plot



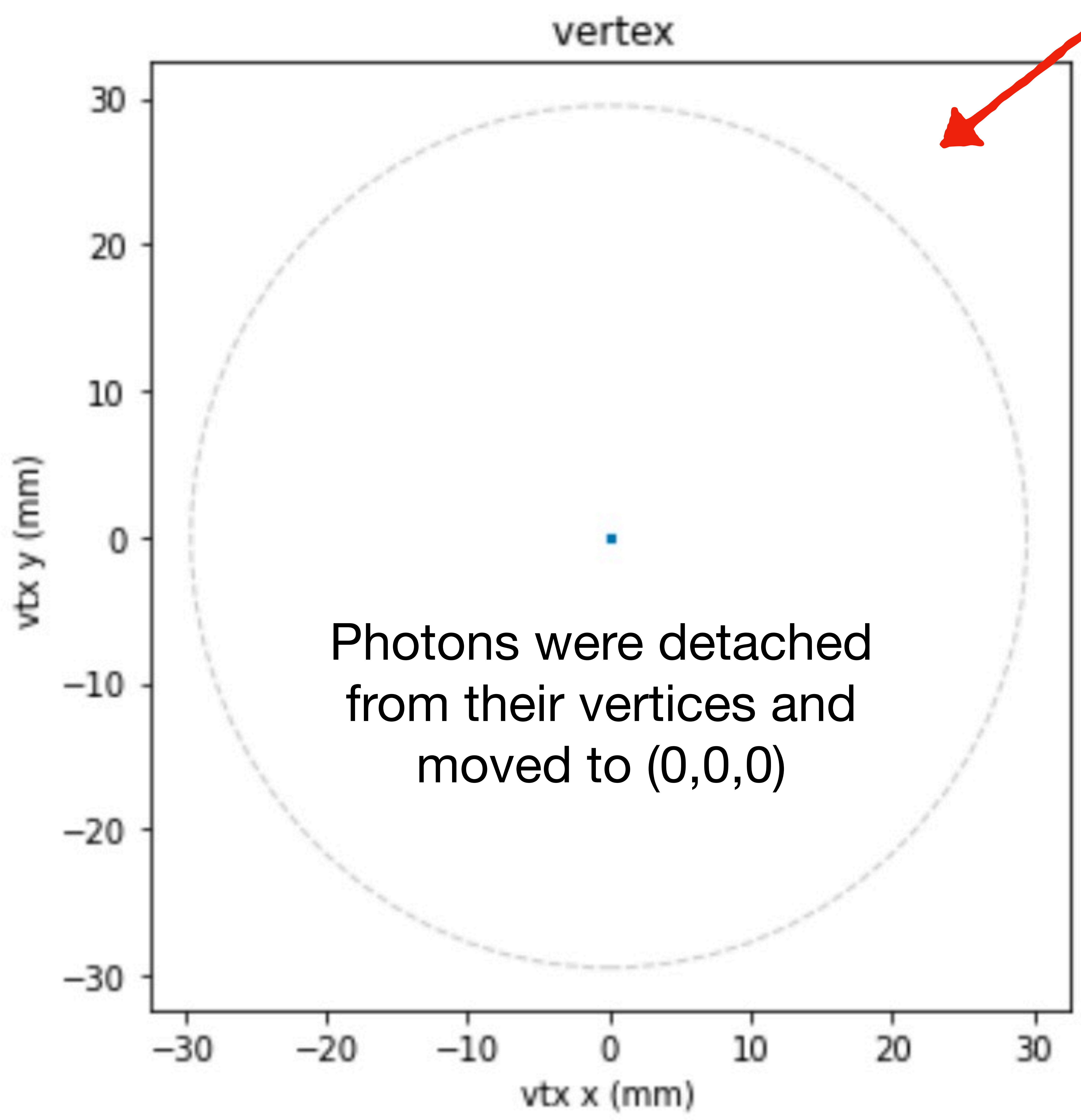
* plots by UC Berkeley undergrad B. Sterwerf

Can we go back now to the original method?

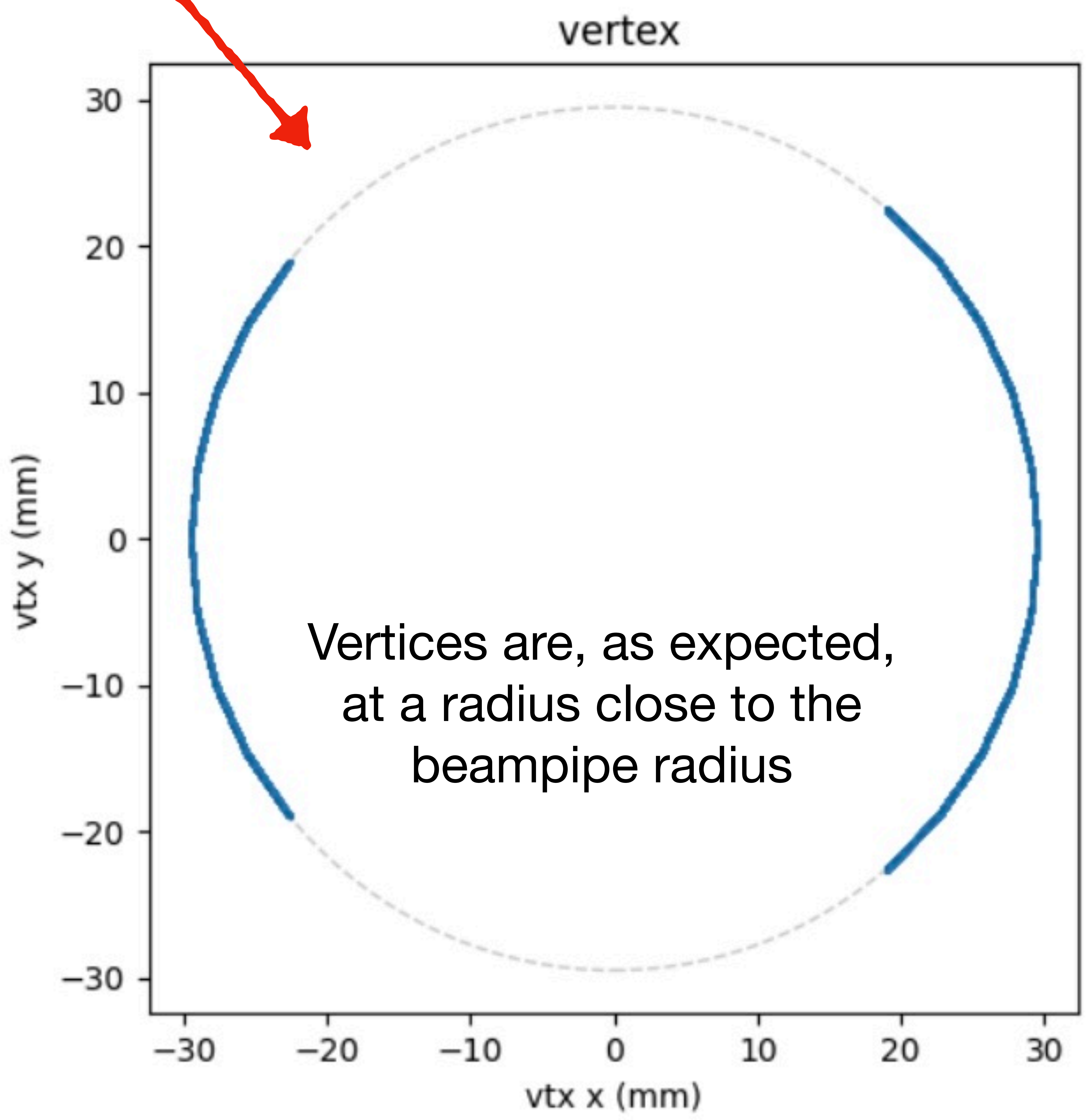


Is the vertex issue in DD4HEP fixed?

Vertex of photons **before** and **after** K. Kauder's fix



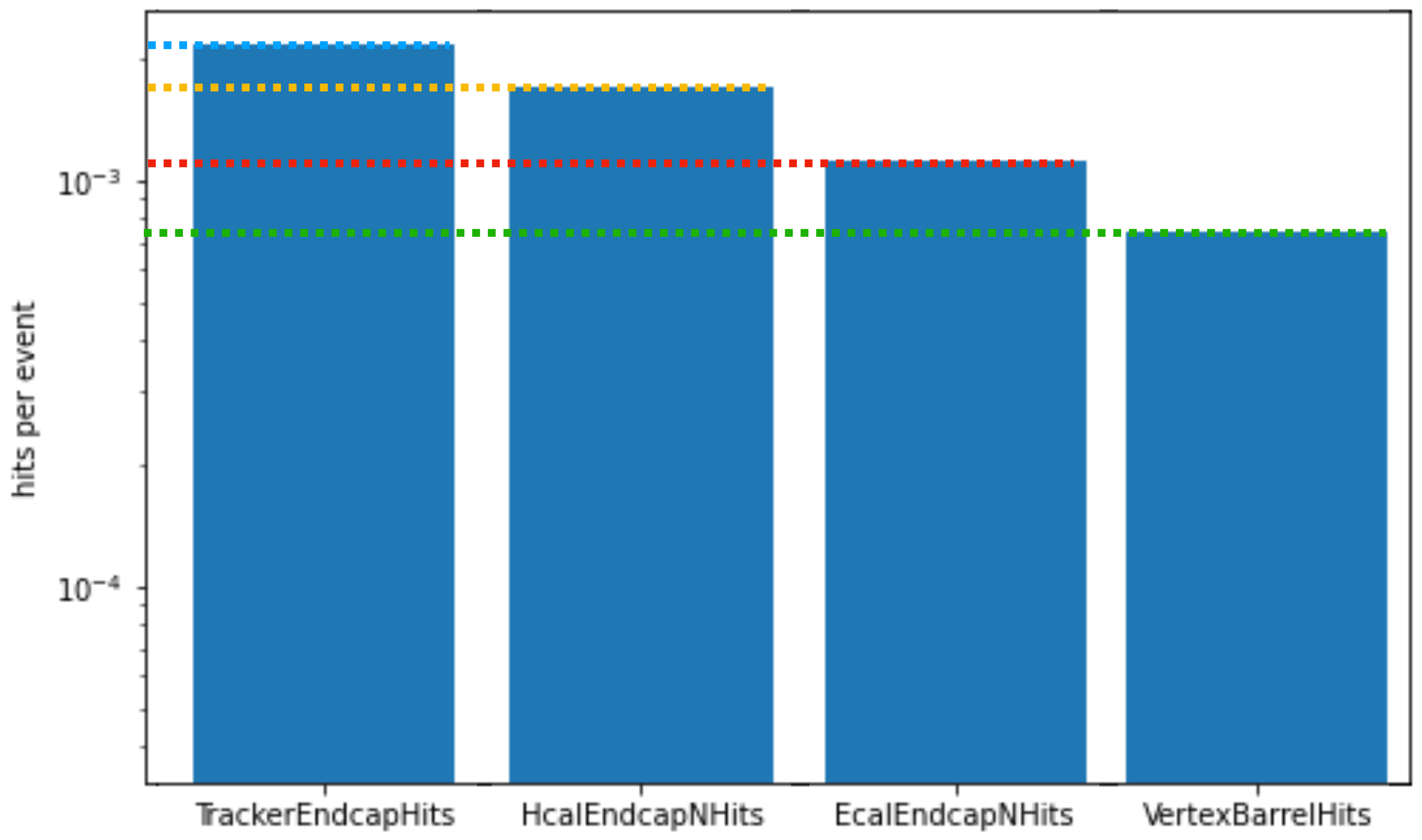
Photons were detached from their vertices and moved to (0,0,0)



Vertices are, as expected, at a radius close to the beampipe radius

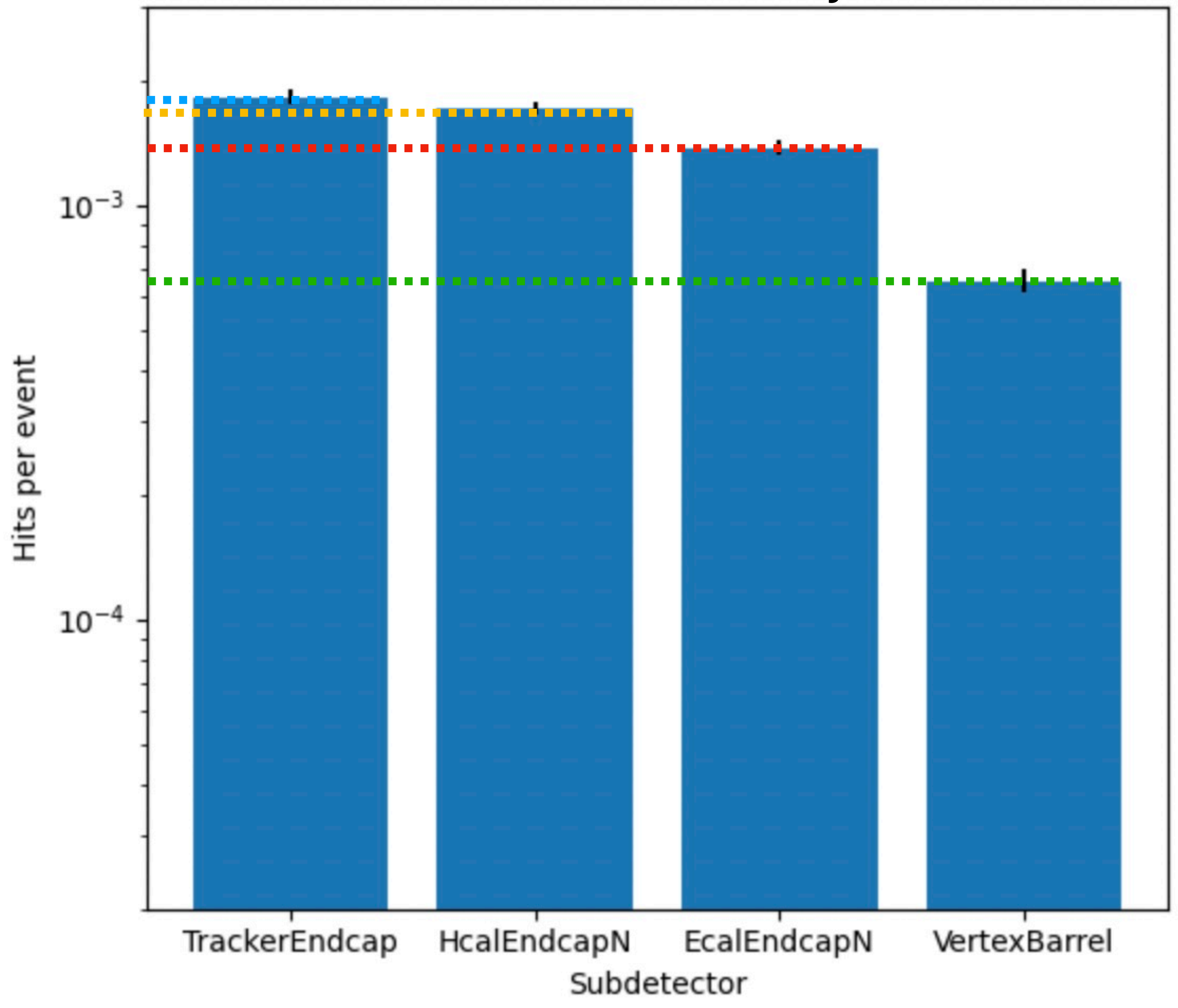
Comparison to Benjamen's results

Alternate (convoluted) method



* plots by UC Berkeley undergrad B. Sterwerf

Standard method after Kolja's vertex fix



flux (for 2.5 A electron beam of E = 10 GeV)

100-ns-wide events

Old (better) method, new results

flux (for 2.5 A electron beam of E = 10 GeV)

100-ns-wide events

Hits per event
10^{-3}
10^{-4}
10^{-5}
TrackerEndcap
HcalEndcapN
EcalEndcapN
EcalBarrelSciGlass
VertexBarrel
TOFBarrel
SiBarrel
MPGDBarrel
EcalEndcapP
B0Tracker
HcalBarrel
LumiSpecCAL
Subdetector

Hits in the tracker disks

flux (for 2.5 A electron beam of E = 10 GeV)

100-ns-wide events

0.0006
0.0005
0.0004
0.0003
0.0002
0.0001
0.0000
Hits per event
−1200
−1000
−800
−600
−400
−200
0
200
400
z [mm]

Hits in the tracker barrel

flux (for 2.5 A electron beam of E = 10 GeV)

100-ns-wide events

Hits per event
0.00040
0.00035
0.00030
0.00025
0.00020
0.00015
0.00010
0.00005
0.00000
50
100
150
200
250
300
350
400
450
$\rho = \sqrt{x^2 + y^2}$ [mm]

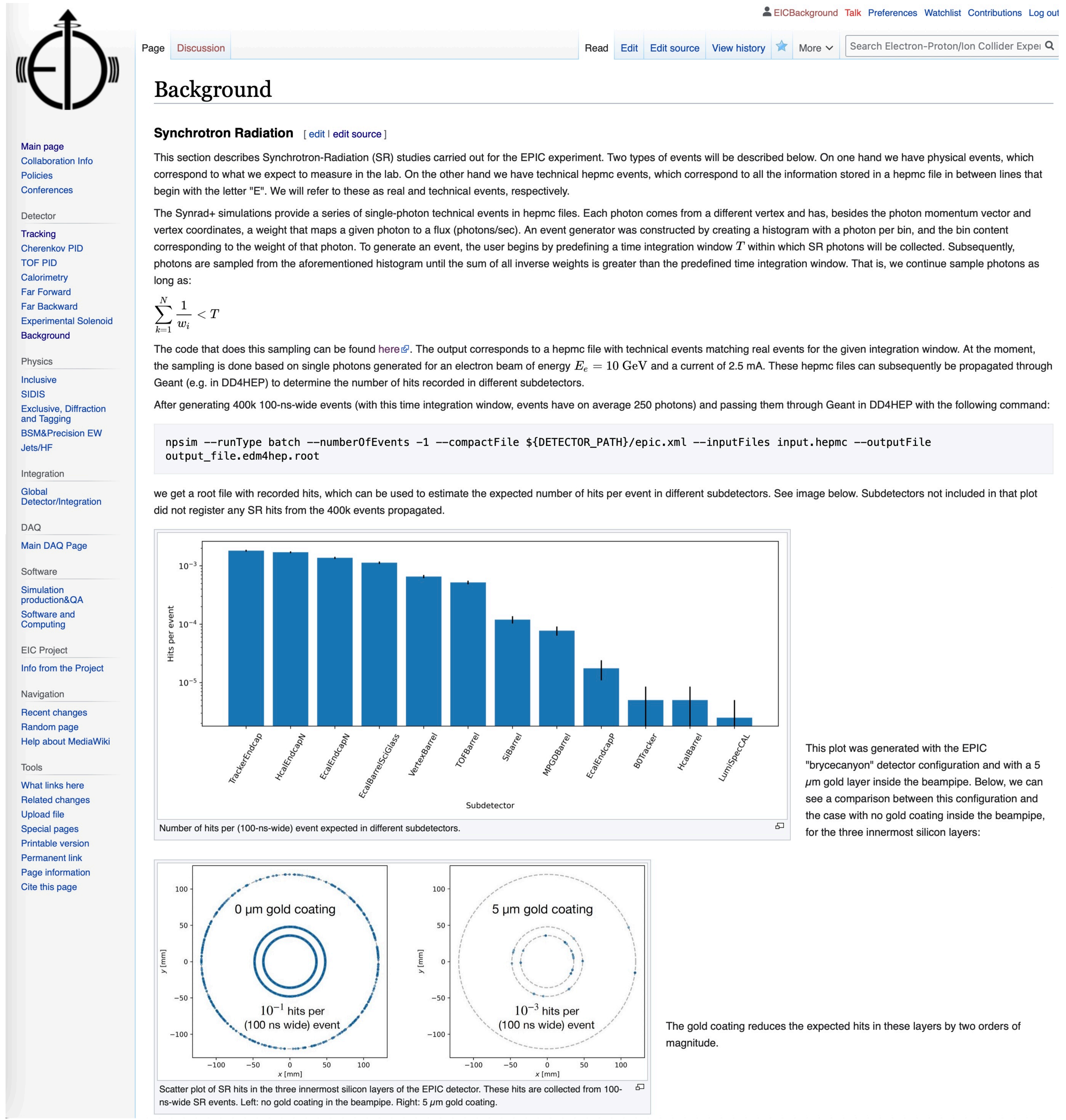
Hits in the tracker barrel

flux (for 2.5 A electron beam of E = 10 GeV)

100-ns-wide events

Hits per event
0.00040
0.00035
0.00030
0.00025
0.00020
0.00015
0.00010
0.00005
0.00000
50
100
150
200
250
300
350
400
450
$\rho = \sqrt{x^2 + y^2}$ [mm]
y
z

Wiki page



Background

Synchrotron Radiation [edit | edit source]

This section describes Synchrotron-Radiation (SR) studies carried out for the EPIC experiment. Two types of events will be described below. On one hand we have physical events, which correspond to what we expect to measure in the lab. On the other hand we have technical hepmc events, which correspond to all the information stored in a hepmc file in between lines that begin with the letter "E". We will refer to these as real and technical events, respectively.

The Synrad+ simulations provide a series of single-photon technical events in hepmc files. Each photon comes from a different vertex and has, besides the photon momentum vector and vertex coordinates, a weight that maps a given photon to a flux (photons/sec). An event generator was constructed by creating a histogram with a photon per bin, and the bin content corresponding to the weight of that photon. To generate an event, the user begins by predefining a time integration window T within which SR photons will be collected. Subsequently, photons are sampled from the aforementioned histogram until the sum of all inverse weights is greater than the predefined time integration window. That is, we continue sample photons as long as:

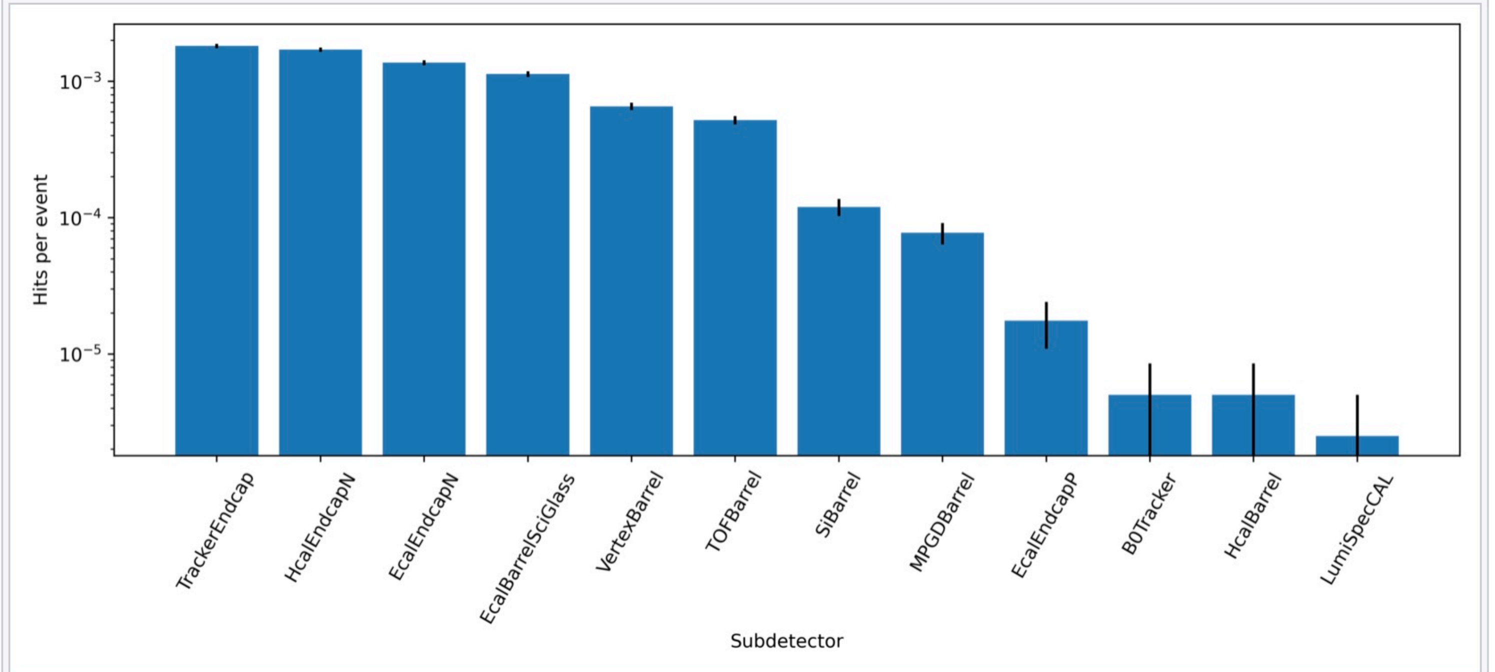
$$\sum_{k=1}^{N} \frac{1}{w_i} < T$$

The code that does this sampling can be found here. The output corresponds to a hepmc file with technical events matching real events for the given integration window. At the moment, the sampling is done based on single photons generated for an electron beam of energy $E_e = 10\ \mathrm{GeV}$ and a current of 2.5 mA. These hepmc files can subsequently be propagated through Geant (e.g. in DD4HEP) to determine the number of hits recorded in different subdetectors.

After generating 400k 100-ns-wide events (with this time integration window, events have on average 250 photons) and passing them through Geant in DD4HEP with the following command:

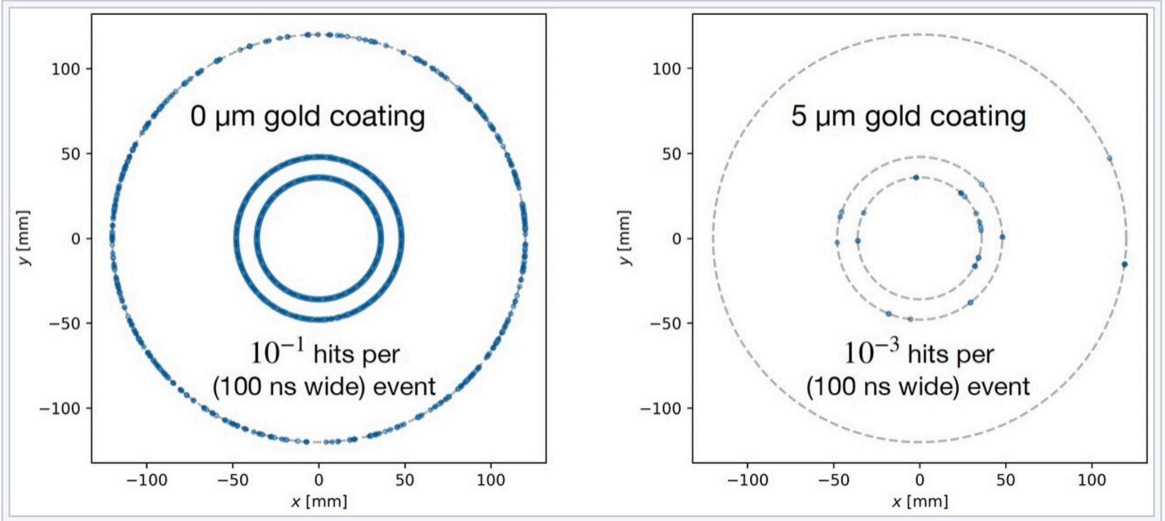
```
npsim --runType batch --numberOfEvents -1 --compactFile ${DETECTOR_PATH}/epic.xml --inputFiles input.hepmc --outputFile
output_file.edm4hep.root
```

we get a root file with recorded hits, which can be used to estimate the expected number of hits per event in different subdetectors. See image below. Subdetectors not included in that plot did not register any SR hits from the 400k events propagated.



Number of hits per (100-ns-wide) event expected in different subdetectors.

This plot was generated with the EPIC "brycecanyon" detector configuration and with a 5 μm gold layer inside the beampipe. Below, we can see a comparison between this configuration and the case with no gold coating inside the beampipe, for the three innermost silicon layers:



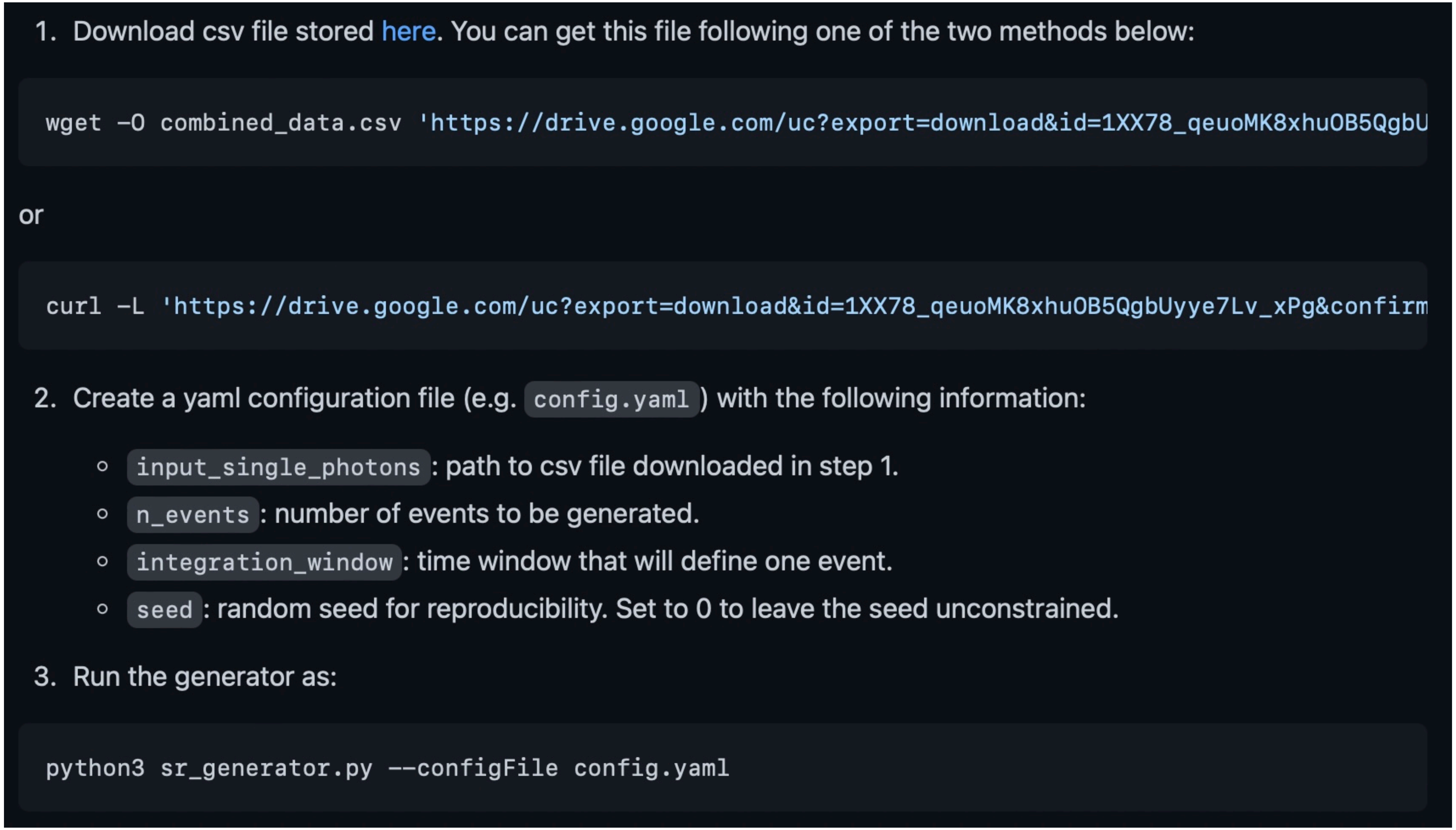
Scatter plot of SR hits in the three innermost silicon layers of the EPIC detector. These hits are collected from 100-ns-wide SR events. Left: no gold coating in the beampipe. Right: 5 μm gold coating.

The gold coating reduces the expected hits in these layers by two orders of magnitude.

Backup

SR event generator

https://github.com/reynier0611/SR_event_generator



1. Download csv file stored here. You can get this file following one of the two methods below:

```
wget -O combined_data.csv 'https://drive.google.com/uc?export=download&id=1XX78_qeuoMK8xhuOB5QgbU
```

or

```
curl -L 'https://drive.google.com/uc?export=download&id=1XX78_qeuoMK8xhuOB5QgbUyye7Lv_xPg&confirm
```

2. Create a yaml configuration file (e.g. `config.yaml`) with the following information:
   - `input_single_photons`: path to csv file downloaded in step 1.
   - `n_events`: number of events to be generated.
   - `integration_window`: time window that will define one event.
   - `seed`: random seed for reproducibility. Set to 0 to leave the seed unconstrained.
3. Run the generator as:

```
python3 sr_generator.py --configFile config.yaml
```