



EUROPEAN  
SPALLATION  
SOURCE

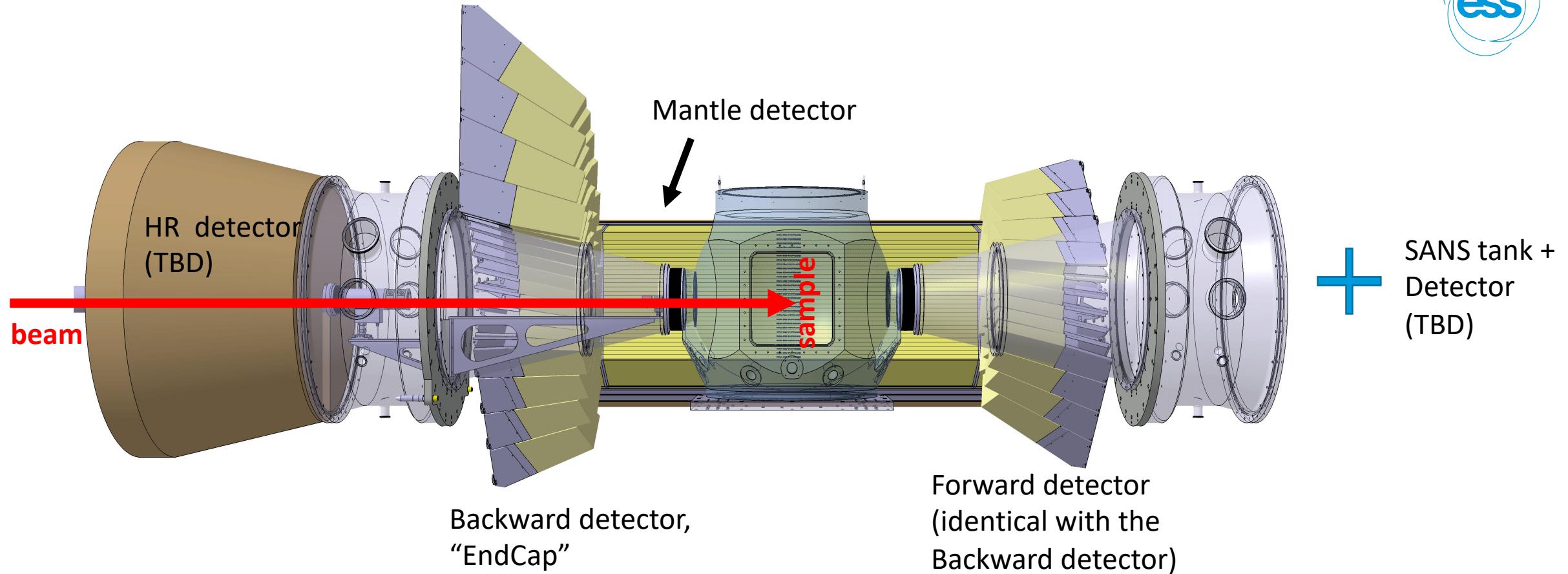


# Channel mapping for the DREAM EndCap detector

PRESENTED BY IRINA STEFANESCU

2020-08-05

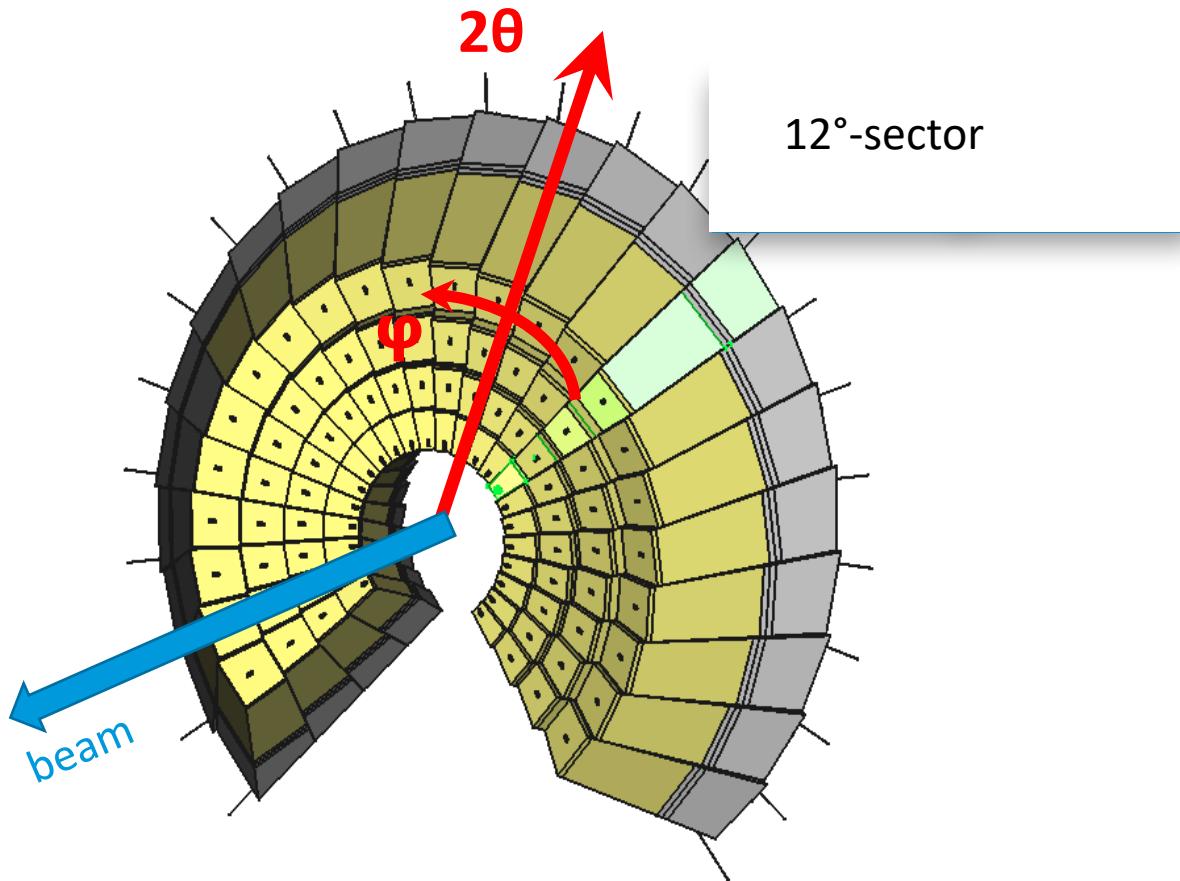
# The DREAM sample scattering system



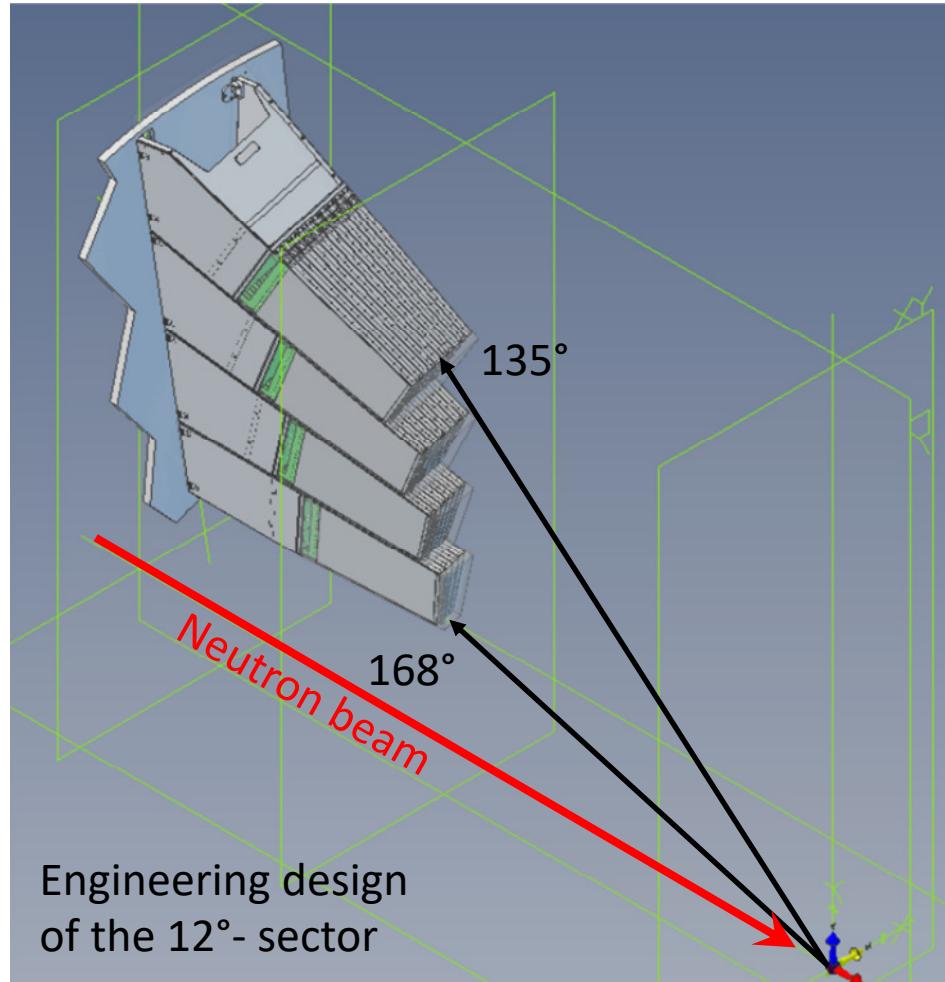
All 4 DREAM detectors will be delivered by the CDT company based in Heidelberg, Germany.

All 4 DREAM detectors will be B-10 based detectors. The mantle and EndCap detectors are from the Jalousie detector family, the design of the High Resolution and SANS detectors is still to be decided.

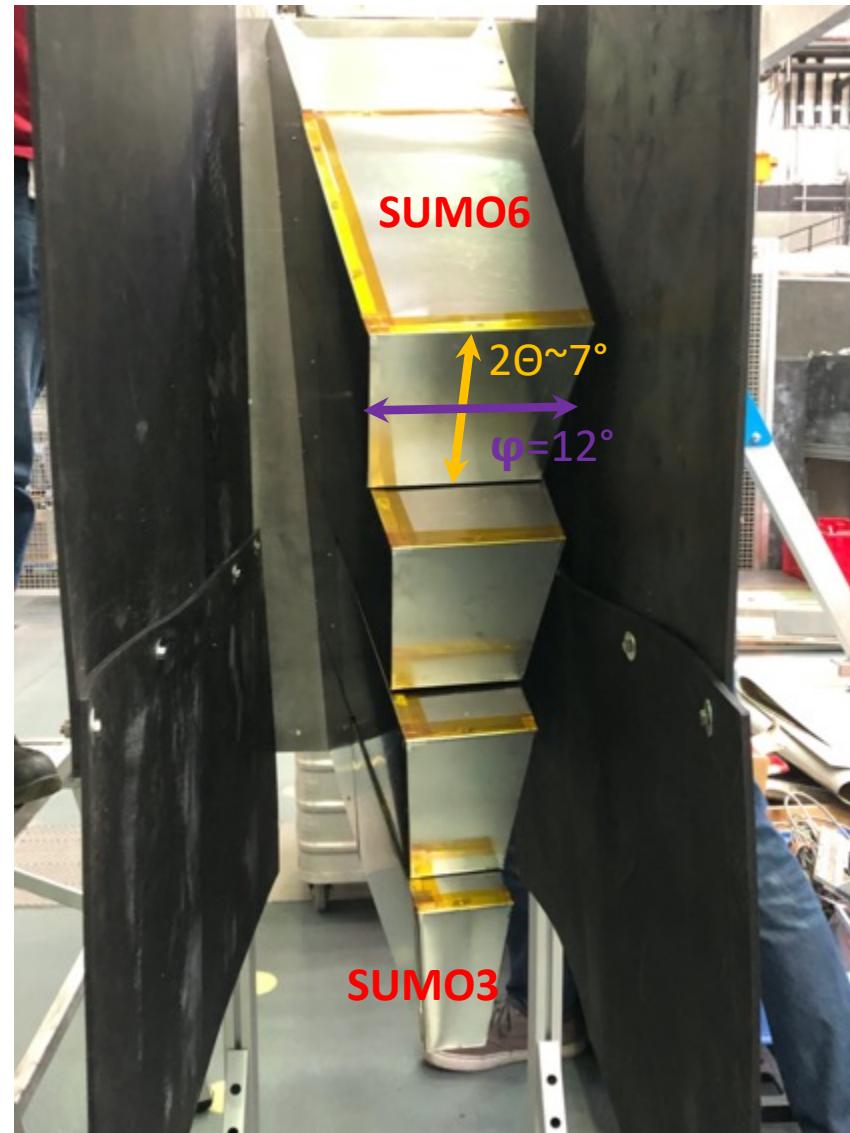
# The DREAM EndCap



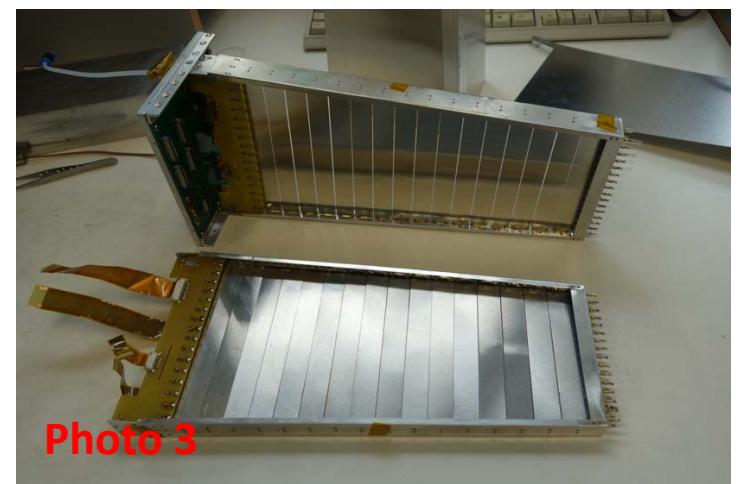
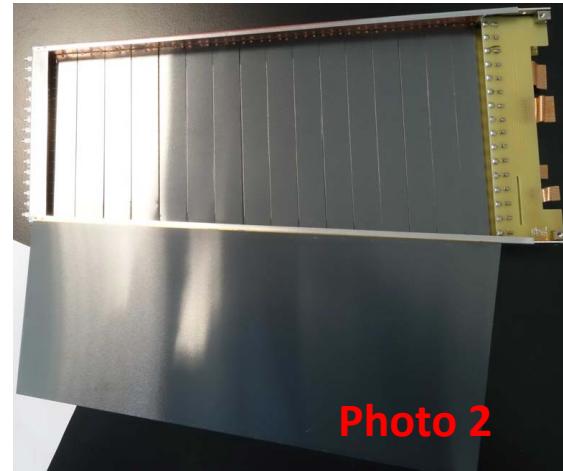
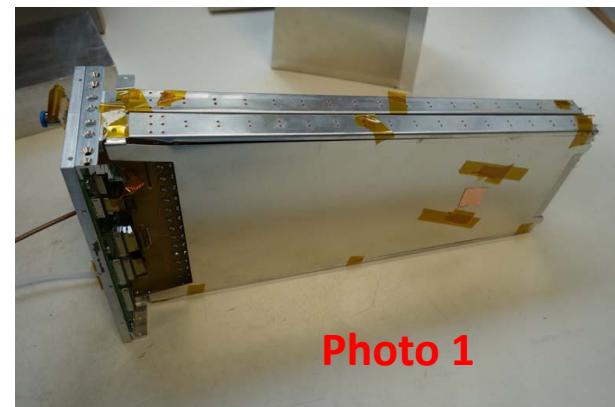
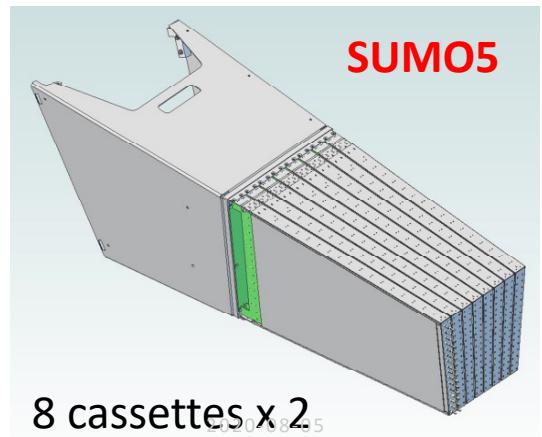
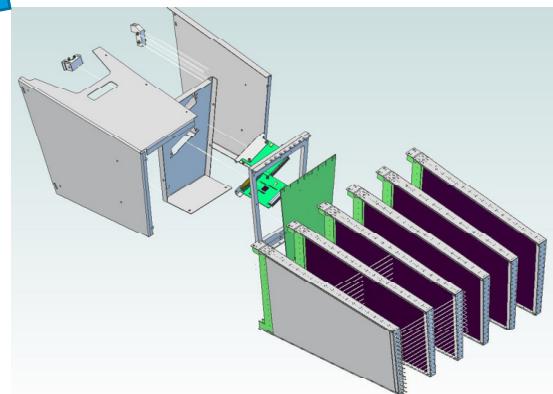
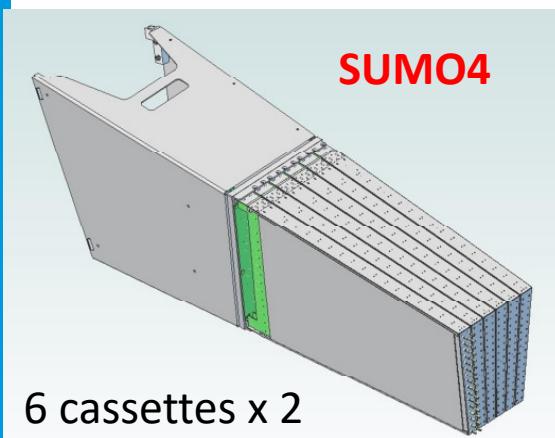
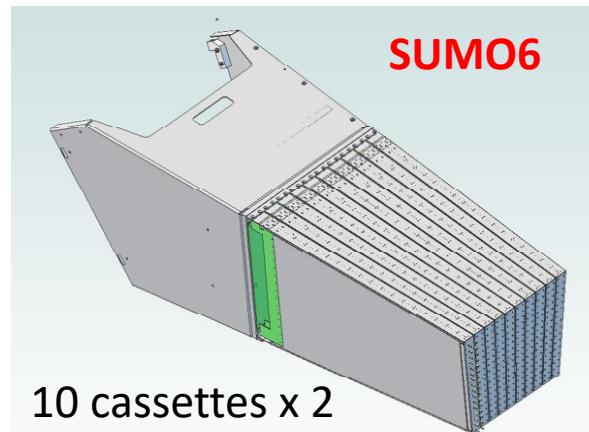
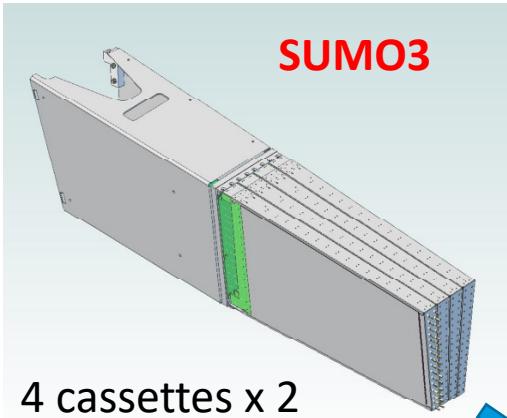
EndCap = assembly of  $12^\circ$ -sectors, each sector consisting of 4 detector modules (SUMOs)

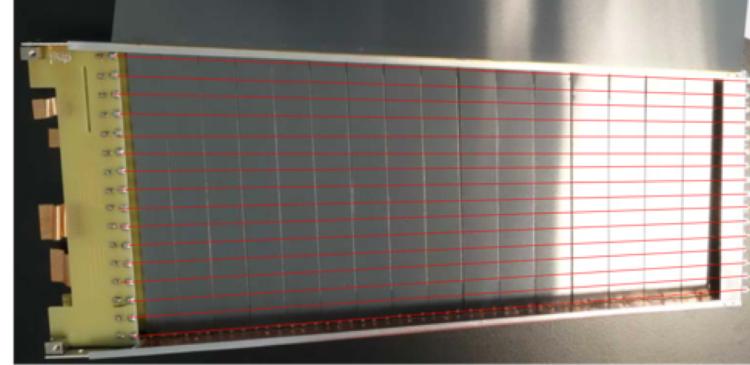
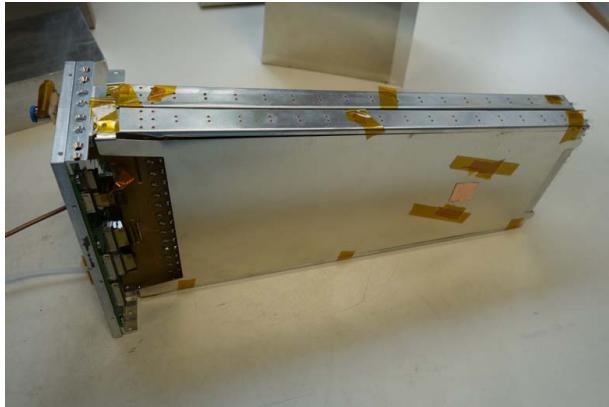


4 SUMO modules (SUper MOdules)  
mounted on the same backplate =  
mounting unit



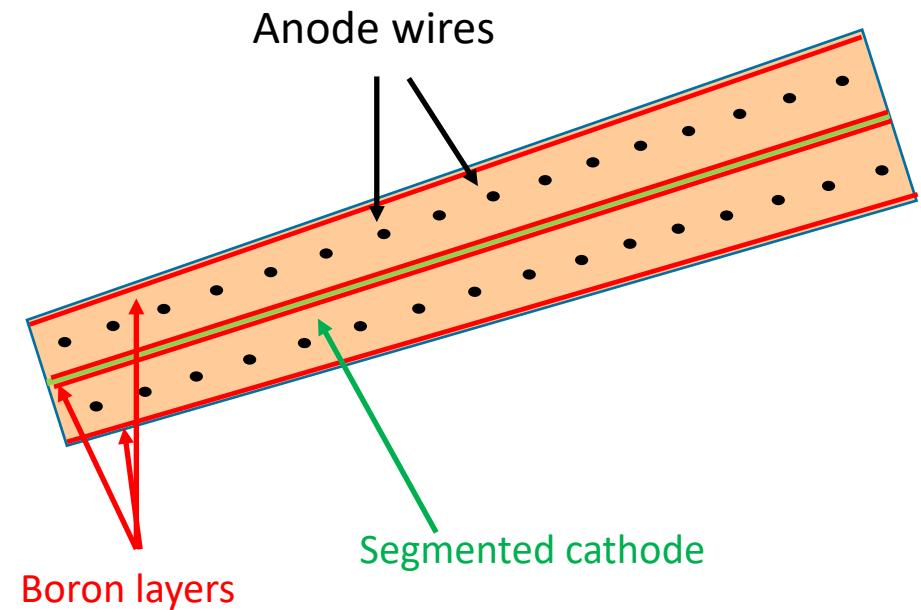
The DREAM 12°- sector @V20





Each segment consists of 2 independent wire counters sharing a common segmented cathode.

16 wires and 16 equal strips in each counter



The wires and the cathode strips are readout independently by 64-channels ASICs



| Detectors for DREAM | No of segments per module | No of ASIC channels per segment | Total number of ASIC channels per SUMO module | Total number of voxels per SUMO module |  |
|---------------------|---------------------------|---------------------------------|---|--|--|
| EndCap              | SUMO3                     | 4                               | $16(w) + 16(s) + 16(w)$                       | $4 \times 3 \times 16 = 192$           | $4 \times 2 \times 16 \times 16 = 2\,048$  |
|                     | SUMO4                     | 6                               | $16(w) + 16(s) + 16(w)$                       | $6 \times 3 \times 16 = 288$           | $6 \times 2 \times 16 \times 16 = 3\,072$  |
|                     | SUMO5                     | 8                               | $16(w) + 16(s) + 16(w)$                       | $8 \times 3 \times 16 = 384$           | $8 \times 2 \times 16 \times 16 = 4\,096$  |
|                     | SUMO6                     | 10                              | $16(w) + 16(s) + 16(w)$                       | $10 \times 3 \times 16 = 480$          | $10 \times 2 \times 16 \times 16 = 5\,120$ |

(w) – wire channel

(s) - strip channel

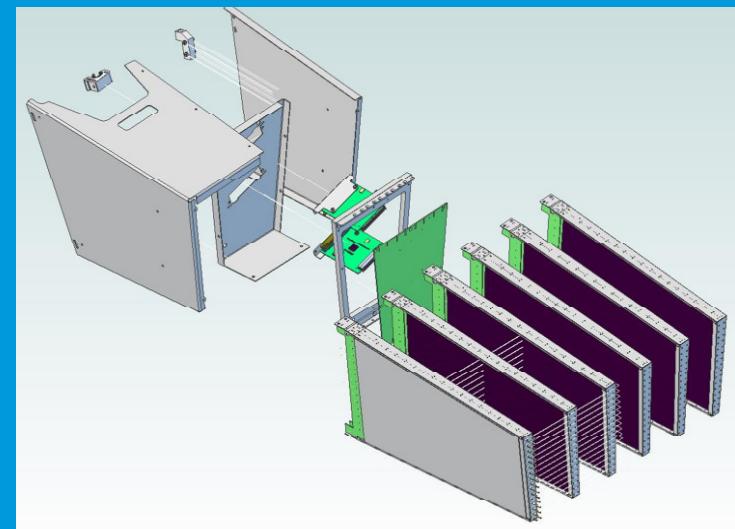
In the raw experimental data the location of a neutron event in the EndCap volume is given by 3 parameters:



- boardID → which SUMO recorded the hit
- AnodeID → which ASIC channel that reads out the wires fired
- CathodeID → which ASIC channel that reads out the cathode strips fired

→ For the EndCap the voxel mapping process is done in 2 steps. The first step is the mapping of the ASIC channels to the detector architecture shown in slides 3-5, in the second step the detector architecture is mapped with the voxel coordinates obtained from another source.

## 1. Mapping the readout channels to the detector architecture



Physical wire grid

ASIC channel# for the wire = 'anodeID' entry  
in the data file

sumo\_voxel\_map\_20190711

ASIC channel# for the cathode strip = 'cathodeID'  
entry in the data file

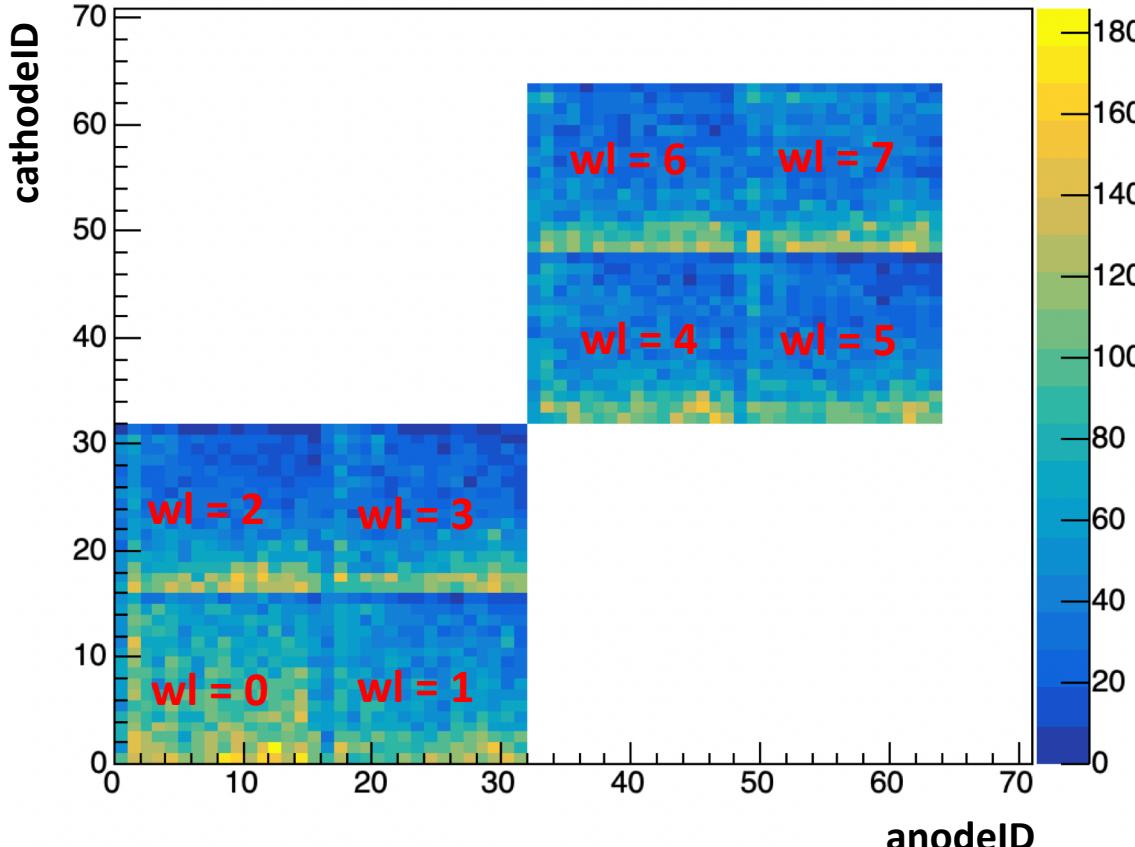
| sumoType | wireLayer | anode | cathode | absoluteXPosition | absoluteYPosition |
|----------|-----------|-------|---------|-------------------|-------------------|
| 3        | 0         | 0     | 0       | 0                 | 0                 |
| 3        | 0         | 0     | 1       | 0                 | 1                 |
| 3        | 0         | 0     | 2       | 0                 | 2                 |
| 3        | 0         | 0     | 3       | 0                 | 3                 |
| 3        | 0         | 0     | 4       | 0                 | 4                 |
| 3        | 0         | 0     | 5       | 0                 | 5                 |
| 3        | 0         | 0     | 6       | 0                 | 6                 |
| 3        | 0         | 0     | 7       | 0                 | 7                 |
| 3        | 0         | 0     | 8       | 0                 | 8                 |
| 3        | 0         | 0     | 9       | 0                 | 9                 |
| 3        | 0         | 0     | 10      | 0                 | 10                |
| 3        | 0         | 0     | 11      | 0                 | 11                |
| 3        | 0         | 0     | 12      | 0                 | 12                |
| 3        | 0         | 0     | 13      | 0                 | 13                |
| 3        | 0         | 0     | 14      | 0                 | 14                |
| 3        | 0         | 1     | 0       | 1                 | 0                 |
| 3        | 0         | 1     | 1       | 1                 | 1                 |
| 3        | 0         | 1     | 2       | 1                 | 2                 |

The information on the wiring of the detector is included in the CSV file sent by CDT in summer 2019:

[sumo\\_voxel\\_map\\_20190711.csv](#)

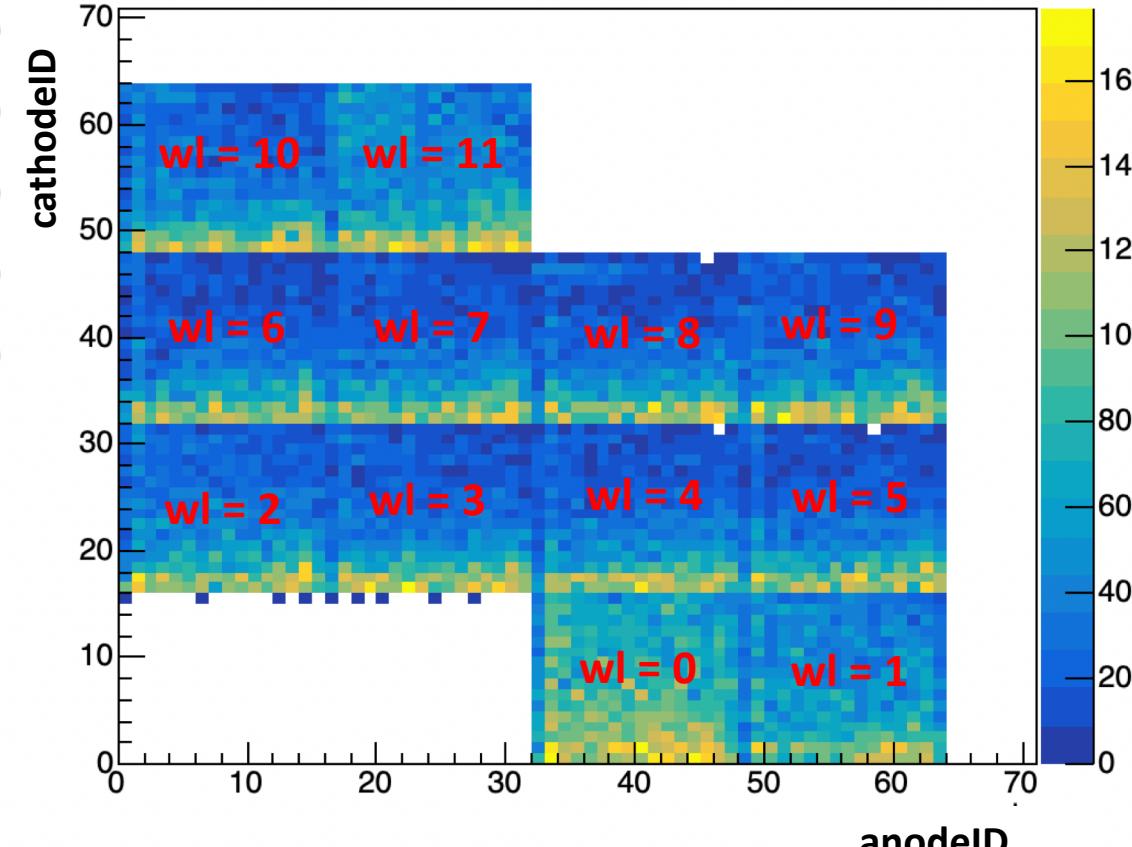
Use these 3 columns to get the plots shown in the next two slides

SUMO3



4 segments x 2 counters/segment = 8 wire layers

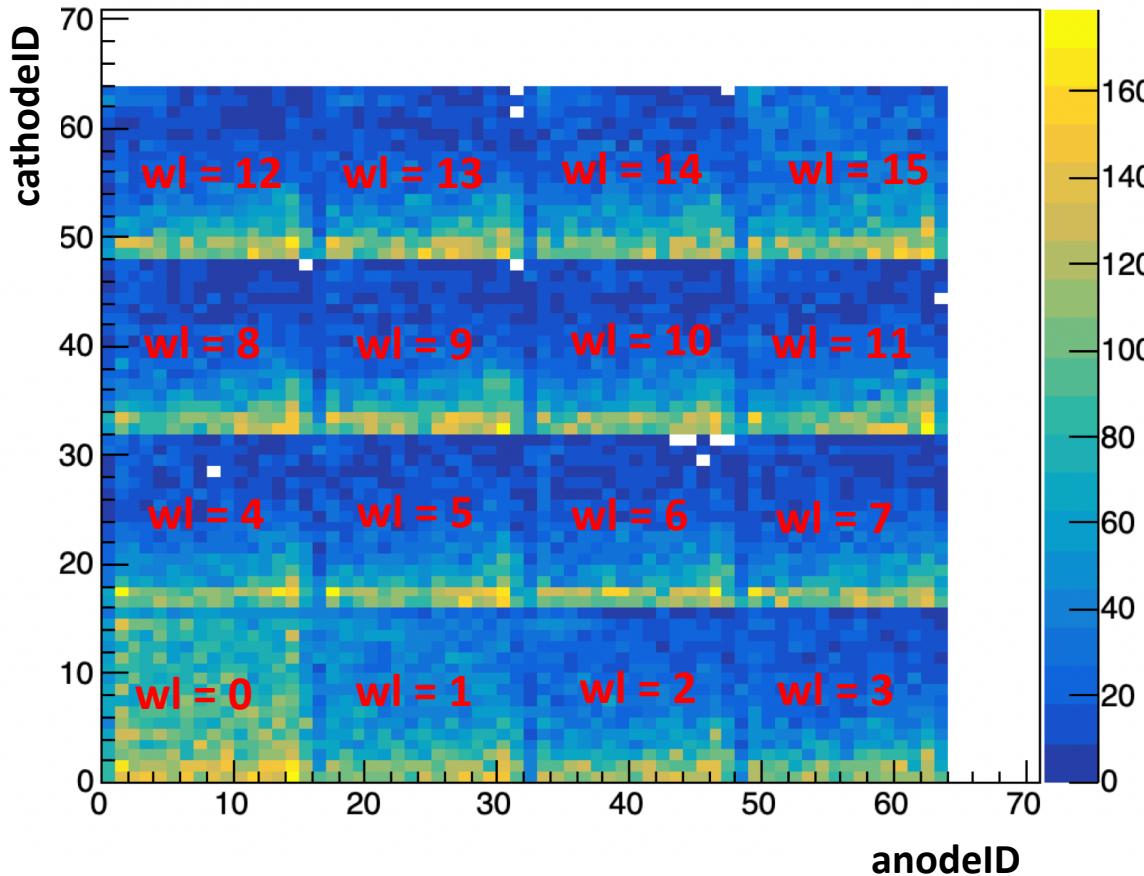
SUMO4



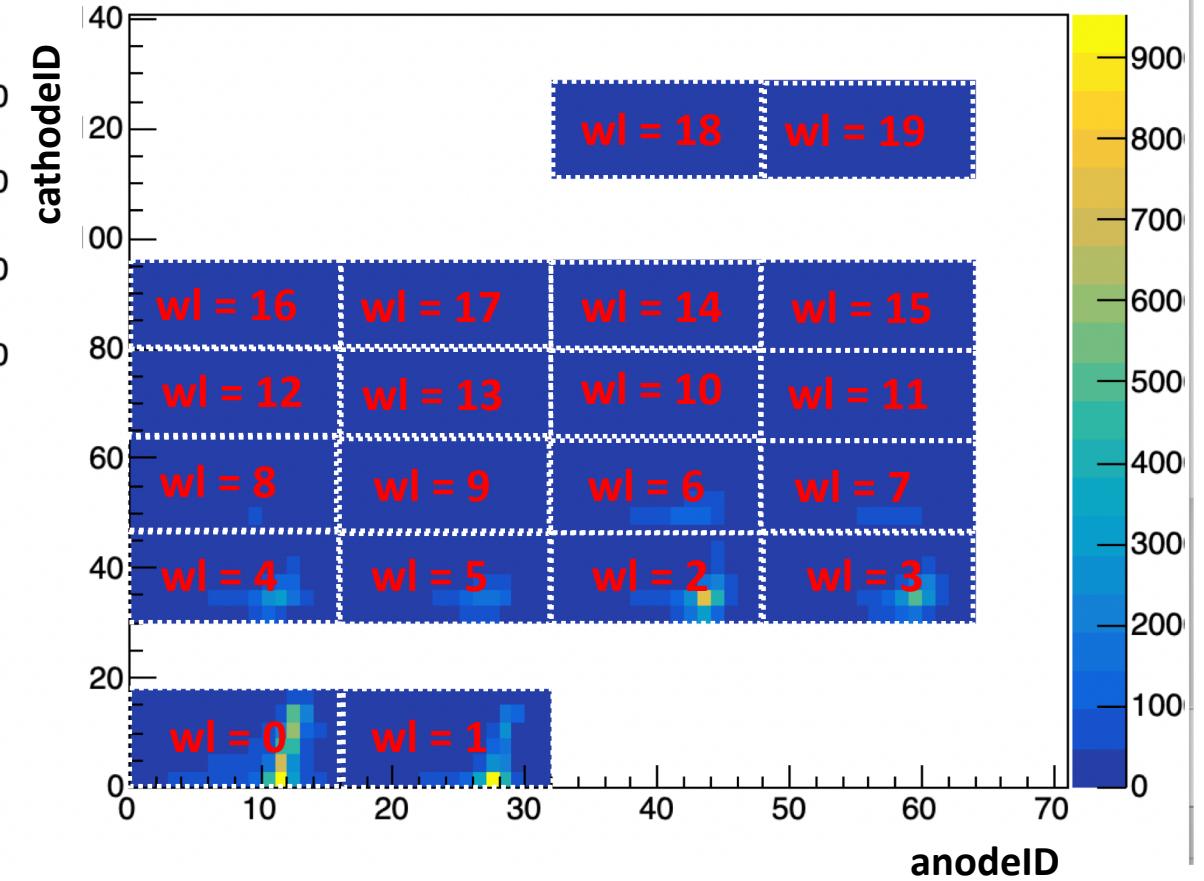
6 segments x 2 counters/segment = 12 wire layers

X-axis = absoluteXPosition (5<sup>th</sup> column in the CDT csv file), Y-axis = absoluteYPosition (6<sup>th</sup> column in the CDT csv file)  
Squares correspond to the wire layer (wl), 2<sup>nd</sup> column in the CDT csv file

## SUMO5



## SUMO6



8 segments x 2 counters/segment = 16 wire layers

10 segments x 2 counters/segment = 20 wire layers

This is how this information is used in the code (`analysis_cdt.cpp`) to convert from ASIC channels and wire layers to user-defined ‘location’ of the neutron event in the detector volume.



The info shown graphically in the last two slides is captured in the code like this:

```
int s3[4][4] = {{0,2,-1,-1}, {1,3,-1,-1}, {-1,-1,4,6}, {-1,-1,5,7}}; //sumo3
int s4[4][4] = {{-1,2,6,10}, {-1,3,7,11}, {0,4,8,-1}, {1,5,9,-1}}; //sumo4
int s5[4][4] = {{0,4,8,12}, {1,5,9,13}, {2,6,10,14}, {3,7,11,15}}; //sumo5
int s6[4][8] = {{0,-1,4,8,12,16,-1,-1},{1,-1,5,9,13,17,-1,-1},{-1,-1,2,6,10,14,-1,18},{-1,-1,3,7,11,15,-1,19}}; //sumo6
```

Wire layer vectors

For-loop through the neutron events

```
factor_a = floor(ncathode/16);
factor_b = floor(nanode/16);

switch (nsumo){
    case 3:
        nw_layer = s3[factor_b][factor_a];
        break;

    case 4:
        nw_layer = s4[factor_b][factor_a];
        break;

    case 5:
        nw_layer = s5[factor_b][factor_a];
        break;

    case 6:
        nw_layer = s6[factor_b][factor_a];
        break;
}

nsegment = floor(nw_layer/2) + 1;
ncounter = nw_layer % 2 + 1;

nwire = nanode - 16*factor_b + 1;
nstrip = ncathode - 16*factor_a + 1;
```

In my analysis code the location of the neutron event in the detector volume is specified by:

- The module (12-deg sector) that was hit (always 1 in this case)
- The SUMO of the module that was hit
- The counter number of the segment that was hit
- The segment number of the SUMO that was hit
- The wire number that collected the charge
- The strip number that collected the charge

Example for SUMO3 which consists of 4 segments and each segment consists of 2 independent wire counters.



```
factor_a = floor(ncathode/16);
factor_b = floor(nanode/16);

switch (nsumo){

    case 3:
        nw_layer = s3[factor_b][factor_a];
        break;

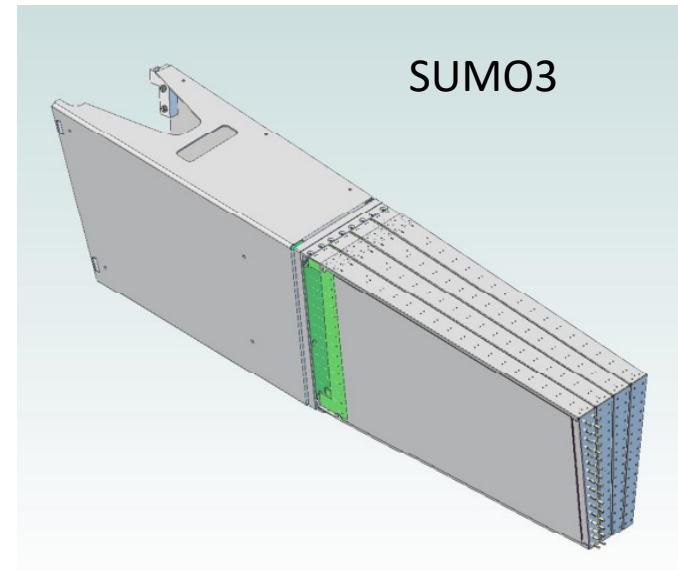
    case 4:
        nw_layer = s4[factor_b][factor_a];
        break;

    case 5:
        nw_layer = s5[factor_b][factor_a];
        break;

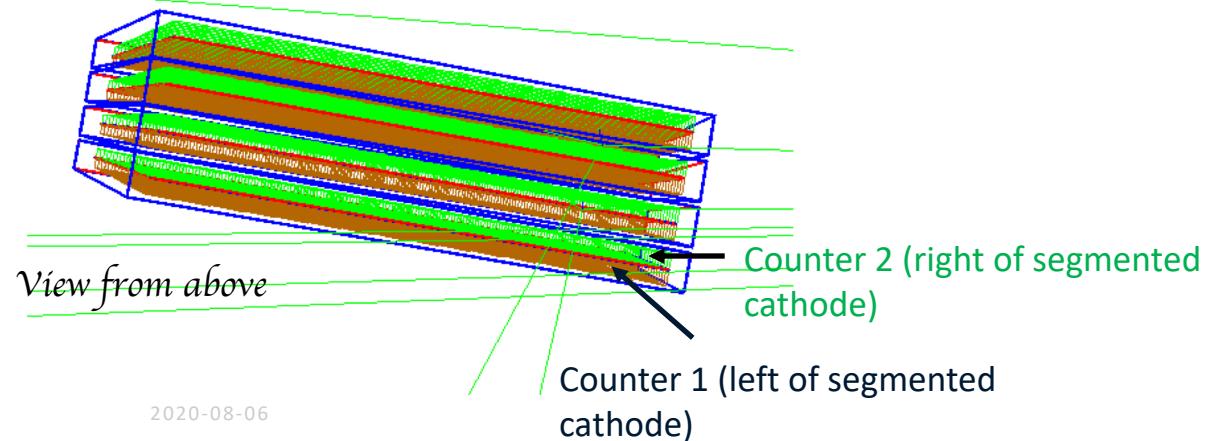
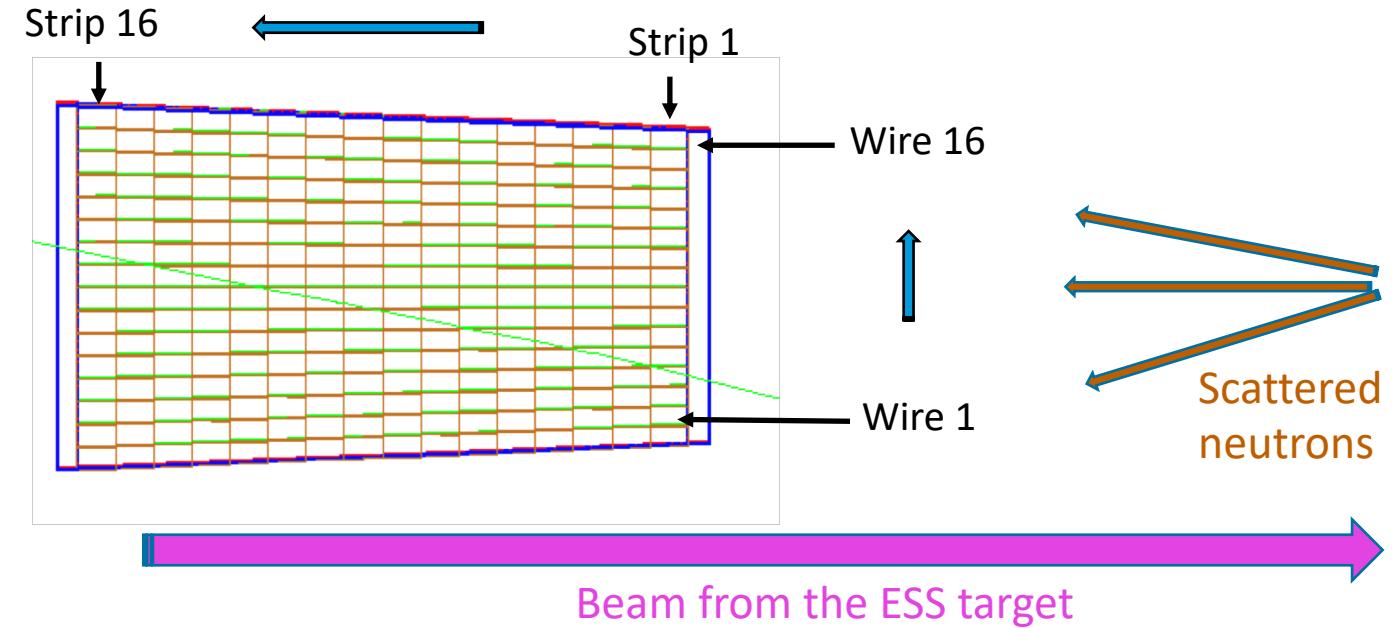
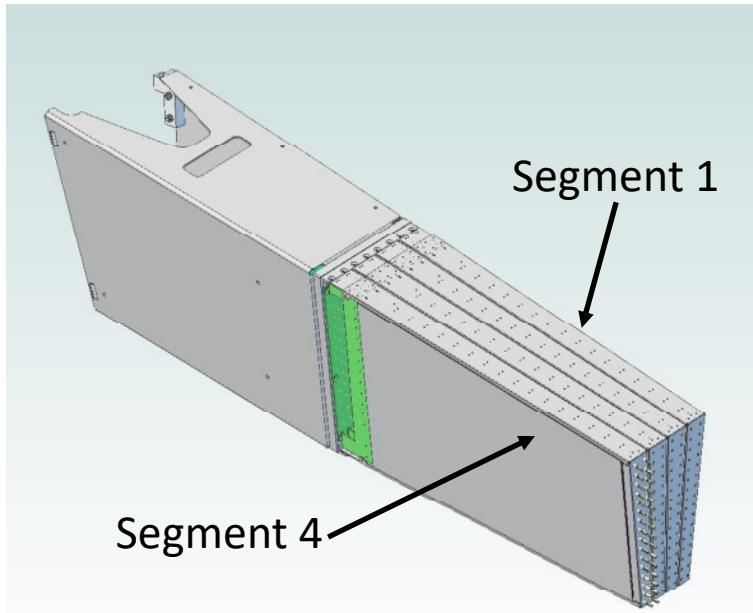
    case 6:
        nw_layer = s6[factor_b][factor_a];
        break;

}

nsegment = floor(nw_layer/2) + 1; → nsegment = 1,..,4
ncounter = nw_layer % 2 + 1; → ncounter = 1, 2
nwire = nanode - 16*factor_b + 1; → nwire = 1,..,16
nstrip = ncathode - 16*factor_a + 1; → nstrip = 1,..,16
```



# Detector geometry, user defined conventions. Example for SUMO3



2020-08-06

! Voxel number must be uniquely defined as a function of the following: sector#, module#, segment#, counter#, strip#, wire# !

# Time-of-flight information

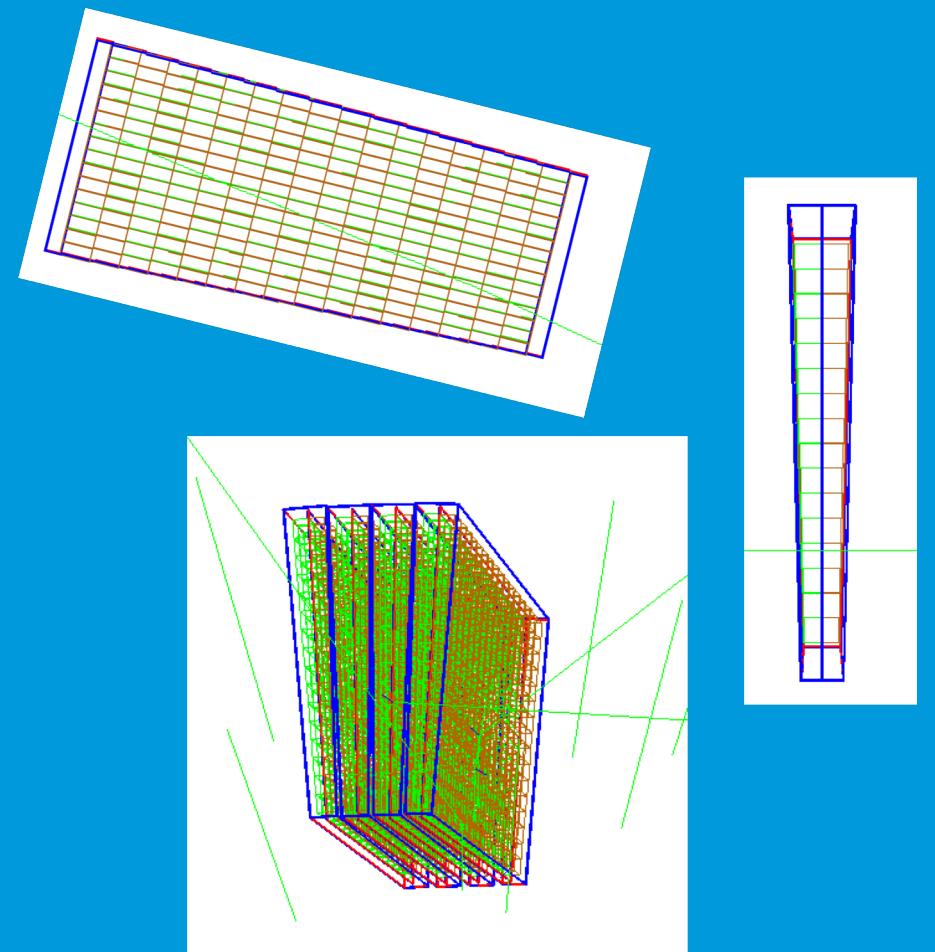
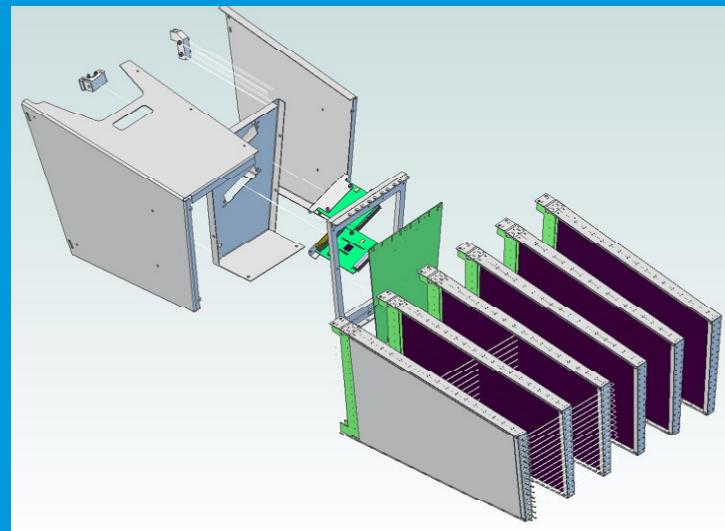


- The neutron time of flight is calculated from the difference of the neutron and chopper timestamps. The time-of-flight obtained from this difference must be corrected in different ways for the runs collected in normal operation mode (full pulse) and Wavelength Frame Multiplication (WFM) mode (the full pulse is split into 6 sub-pulses with the help of a pair of WFM choppers). The time corrections are specific to each instrument (i.e., the time corrections for the V20 data will not be valid for the DREAM instrument.)

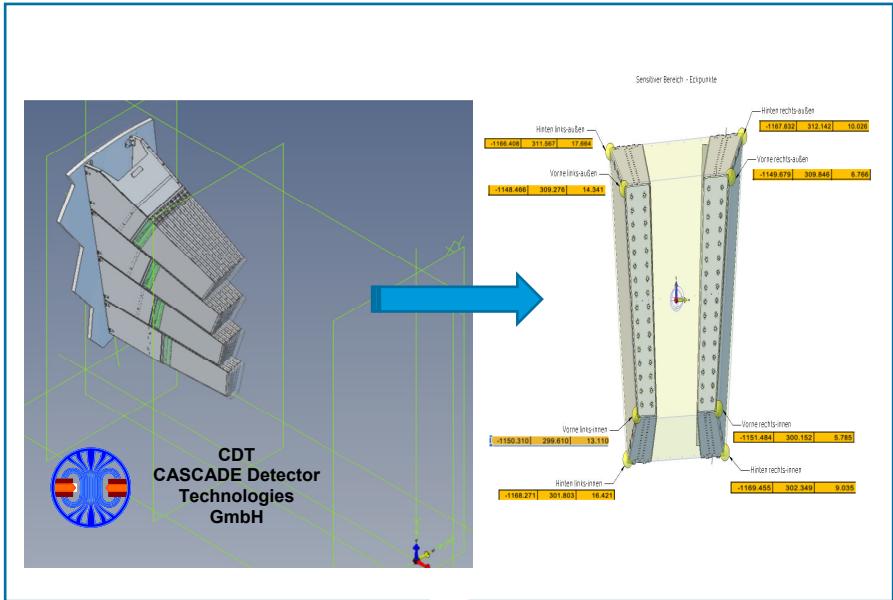
## x,y,z-coordinates of the detector voxels

- Specifying the location of the neutron event in terms of (module, SUMO, segment, counter, wire and strip number) is not enough to perform the transformation from the time-of-flight space to the wavelength- and d-spacing-spaces (according to the Bragg law). In order to calculate the wavelength of the detected neutron one needs the information on the coordinates ( $x$ ,  $y$ ,  $z$ ) of the interaction point with respect to the sample position. Currently there are two ways to do this for the Jalousie data.

## 2. Mapping the detector architecture to voxels coordinates



# Obtaining the coordinates of the centres of the detector voxels, method 1



Link to the CDT Excel file with the CAD coordinates of the SUMO edges:

[https://github.com/ess-dg/dream\\_endcap\\_analysis/blob/refactor/documentation/REAM-vonPOWTEX%20SUMOx-Voxelpositionen26.06.19\\_withtranslationSumo6.xlsx](https://github.com/ess-dg/dream_endcap_analysis/blob/refactor/documentation/REAM-vonPOWTEX%20SUMOx-Voxelpositionen26.06.19_withtranslationSumo6.xlsx)

Obtain the coordinates of the SUMO edges from the CAD engineering drawings

Python script to calculate/interpolate the positions of the voxel centres

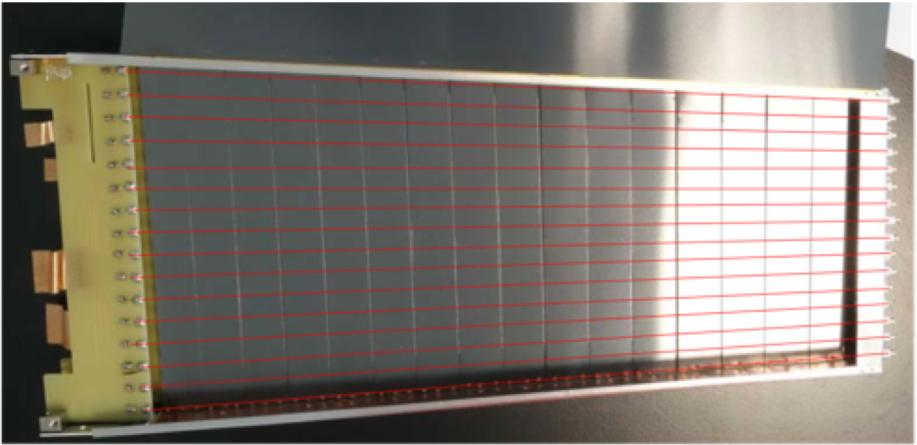
Table with the coordinates of the voxels

Read into the data-analysis code

Jülich data analysis

Done by CDT

Done by FZJ

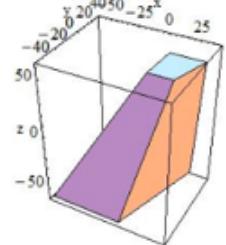


The segments and the gas voxels were hard coded by using the G4Trap class (4 angles and 7 dimensions → hexahedrons).

```
G4Trap(const G4String& pName,
       G4double pZ,
       G4double pY,
       G4double pX,
       G4double pLTX)

G4Trap(const G4String& pName,
       G4double pDz,      G4double pTheta,
       G4double pPhi,     G4double pDy1,
       G4double pDx1,     G4double pDx2,
       G4double pAlp1,    G4double pDy2,
       G4double pDx3,     G4double pDx4,
       G4double pAlp2)
```

*In the picture:*



pDx1 = 30, pDx2 = 40, pDy1 = 40, pDx3 = 10, pDx4 = 14, pDy2 = 16, pDz = 60, pTheta = 20\*Degree, pPhi = 5\*Degree, pAlp1 = pAlp2 = 10\*Degree



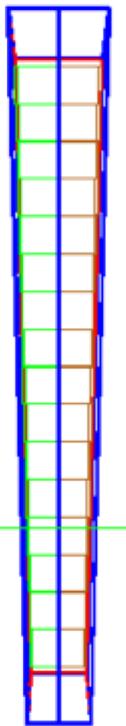
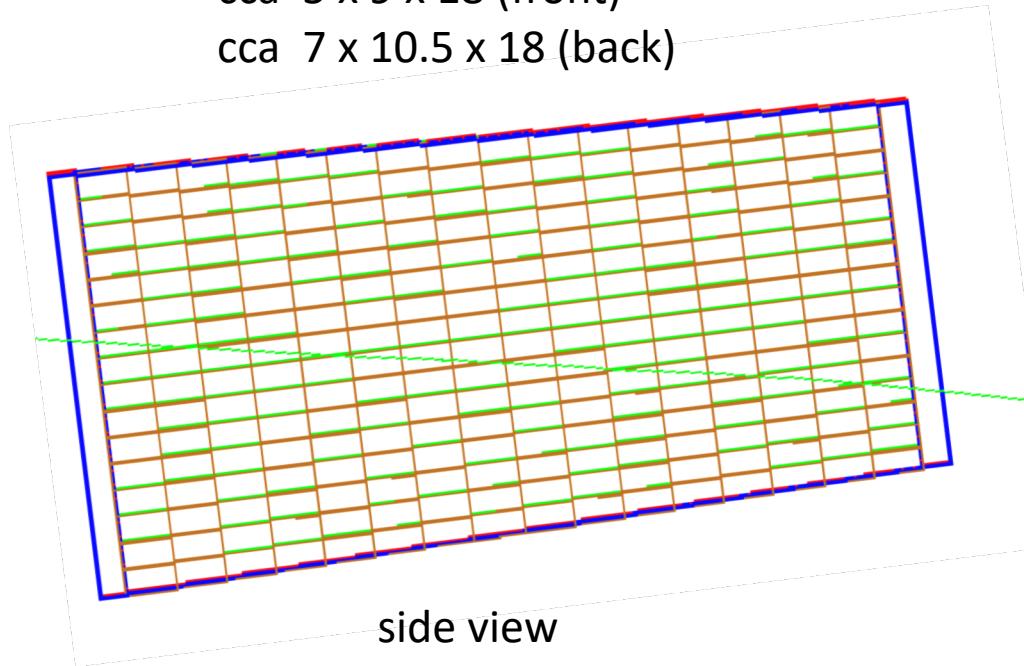
## GEANT4 simulation strategy for the Jalousie detectors



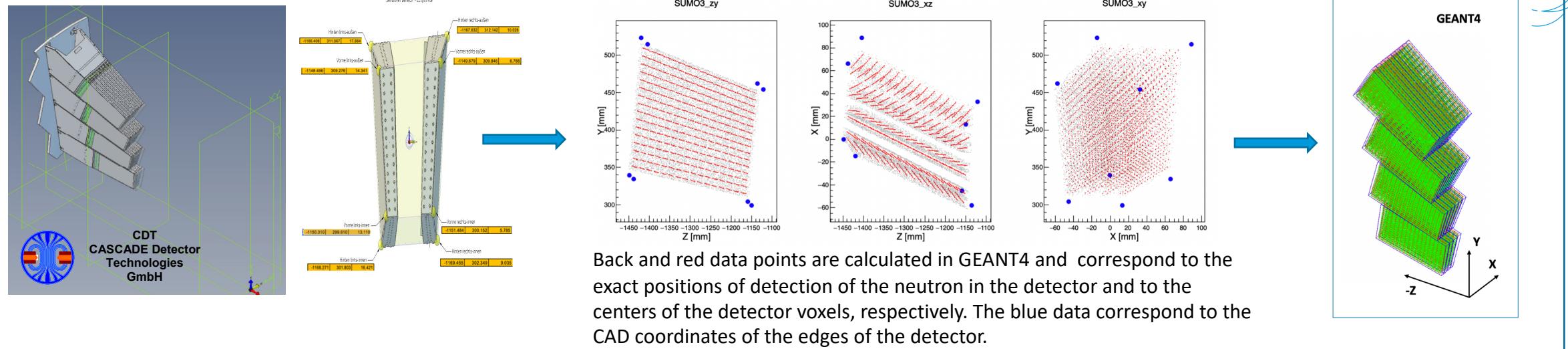
See paper on the G4 model of the HEIMDAL detector:

<https://iopscience.iop.org/article/10.1088/1748-0221/14/10/P10020>

Voxel size (w x h x d) in mm<sup>3</sup>:  
 cca 5 x 9 x 18 (front)  
 cca 7 x 10.5 x 18 (back)



# Obtaining the coordinates of the centres of the detector voxels, method 2



Obtain the coordinates of the SUMO edges from the CAD engineering drawings and match them with the positions of the GEANT4 computer models for the SUMOs (voxel based)



Dump the positions of the voxel centres in a ROOT tree

Done in Lund



Read into the data-analysis code  
Lund data analysis

The voxel coordinates generated by the GEANT4 model of ENDCAP are saved in a ROOT file along with other identifiers, but I can save them in any other format.

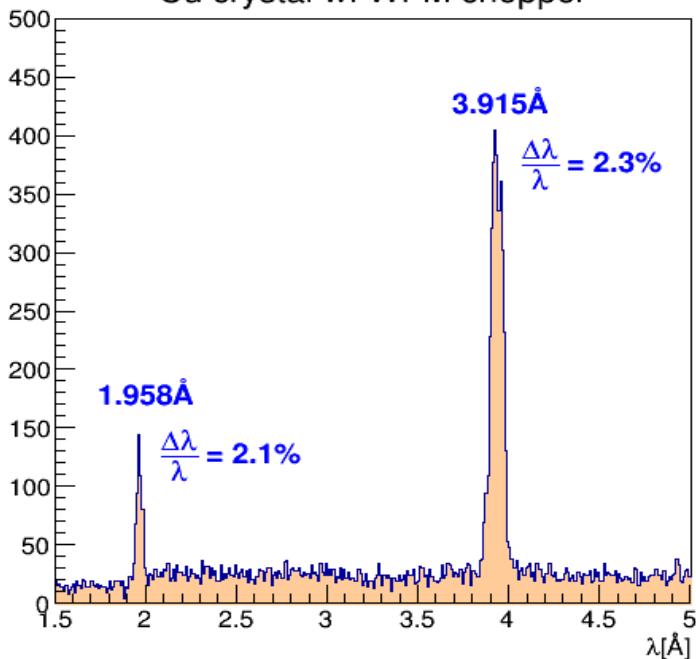


```
irinastefanescu@CI0021360 build % root -l endcap_lookup.root
root [0]
Attaching file endcap_lookup.root as _file0...
(TFile *) 0x7ffab80f38a0
root [1] lookup_tree->Show(10)
=====> EVENT:10
sumo      = 3
wire      = 11
seg       = 1
counter   = 1
strip     = 1
module    = 1
indexi   = 300101
indexj   = 100111
posx     = 25.5
posy     = 404.99
posz     = -1147.14
root [2]
```

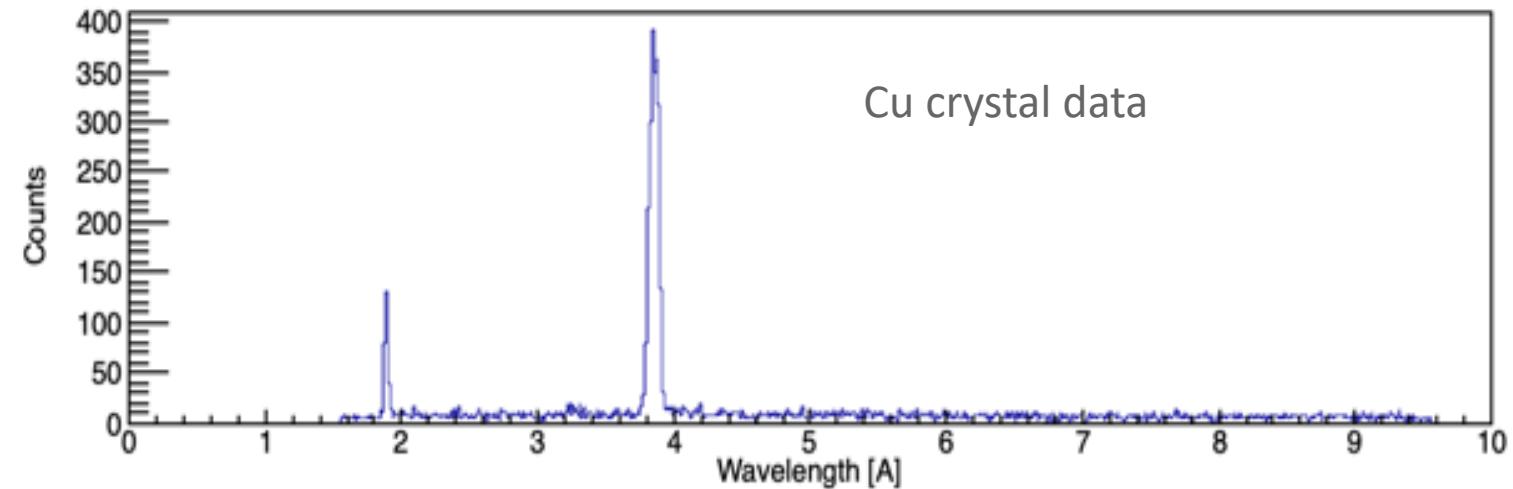
# Example of physics results obtained by using the voxel coordinates from GEANT4



Cu crystal w. WFM chopper



Jülich data analysis & voxel mapping



Lund data analysis & voxel mapping



# Finish presentation

2020-08-05