UNIVERSITY OF BERGEN

“Bergen” analysis of EUtelescope n-tuples

Bjarne Stugu

No big ambitions

- A few years ago I wanted to be able to have a quick look at the data
- Give a starting point for students
- SO WHAT I DID was to use 'MakeClass':
  -
```
// åpne fila
TFile *f = new TFile("NTuple.root")
//  Definere object med datastruktur som i CollectionTree
 TTree *tree = (TTree*)f->Get("CollectionTree")
//To måter å lage skjellettene:
// enkel klasse
tree->MakeClass("f1")
```

MakeClass sniffs out the data structure of a branch and creates the necessary variable definitions in a header file

– To Run from root:

.L clanalyse.C++   (important to compile for speed)

clanalyse t

t.Loop()

Then some development started

- Decode into fixed sized arrays
- Make a generic clustering routine
  - Which was used to check the tbmon clustering
  - Which also works fine on the EUtelescope tracker plane hits that are generated by Andreas with allpix

```
// Hits and Cluster variables
      const int nDts = 5;
      const int maxHits = 400;
      const int maxSz = 151;
      const int maxNcl = 151;
      int  NhMod[nDts],RowDut[nDts][maxHits],ColDut[nDts][maxHits],TotDut[nDts][maxHits];
      int ncl,nused,itotmax,clsz[maxNcl],lclall[maxNcl][maxSz],lclindex[maxSz],cltot[maxNcl],RowD[maxSz],ColD[maxSz],TotD[maxSz],
          notused[maxHits];
      double Xcl[maxNcl][maxSz],Ycl[maxNcl][maxSz],Tcl[maxNcl][maxSz];
      double XcogD[maxNcl],YcogD[maxNcl];
```

Now there is a 'framework'

Tbmon
Ntuple.root

Allpix simulation ascii data file

Cluster hits in tracker planes
'Find' and reconstruct tracks
Make impact point predictions

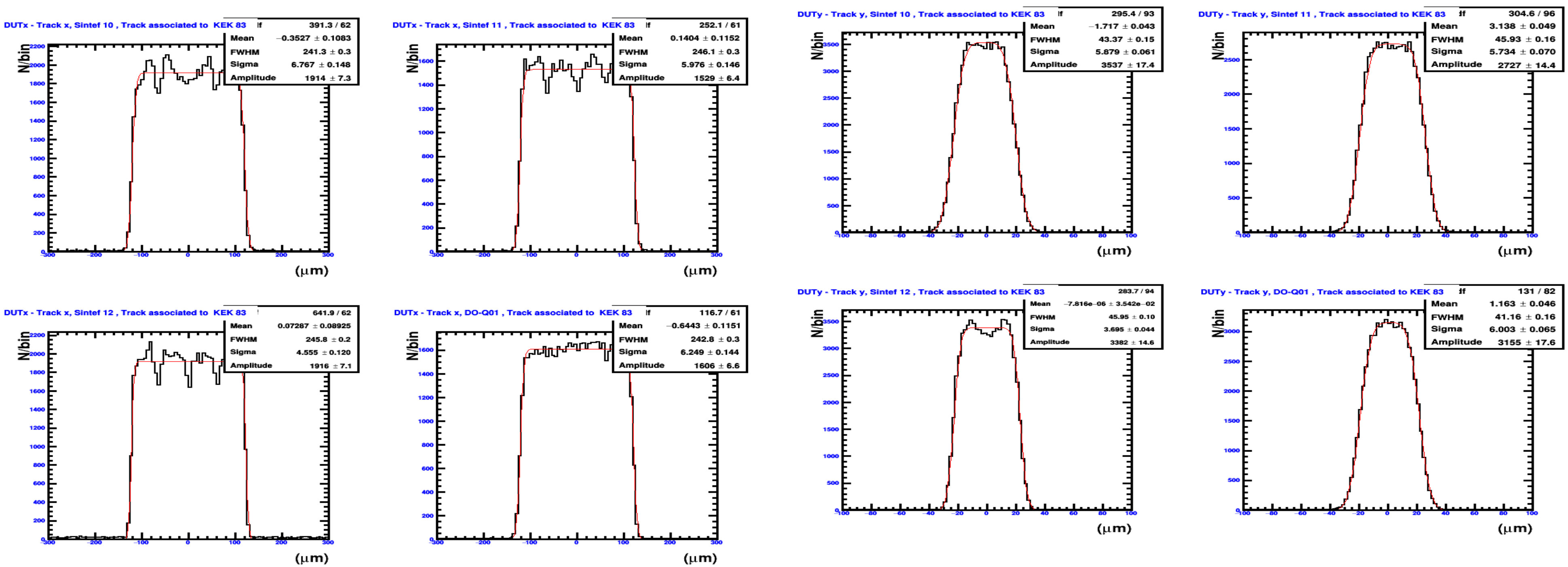
New event file with 'tbmon' information

clanalyse.C
Decodes information
clustering
Does analysis

The efficiency analysis

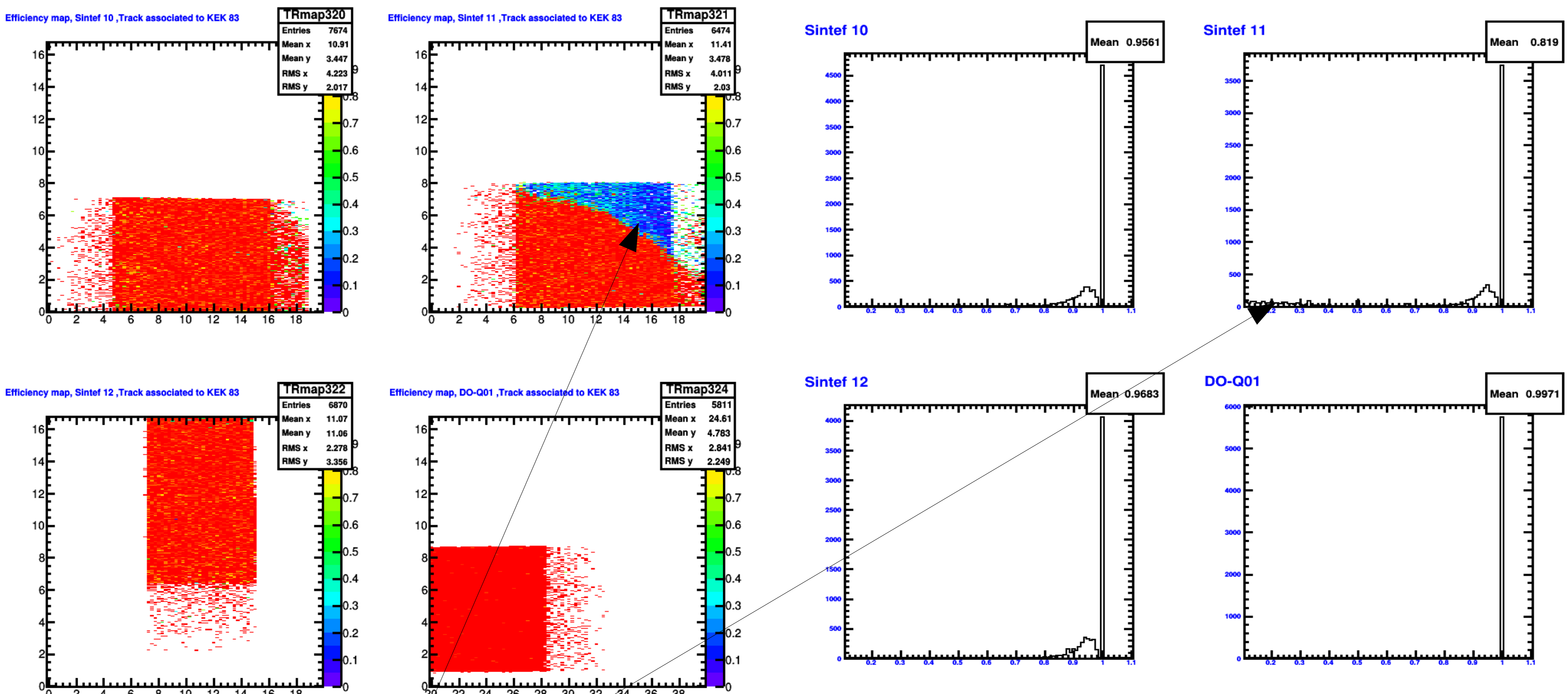
- Get run number
  - Get alignment corrections
  - Get track-hit association window sizes
    - For association to a reference sensor
    - For association to the DUTs
  - Define reference sensor
    - The rest are DUTs
- Timing
  - 1min 40 sec for 5.7 Mevents

Alignment/Resoluton



Impact resolution:
5.5-6 microns for SIntef10 and 11
3.5-4 microns for Sintef12

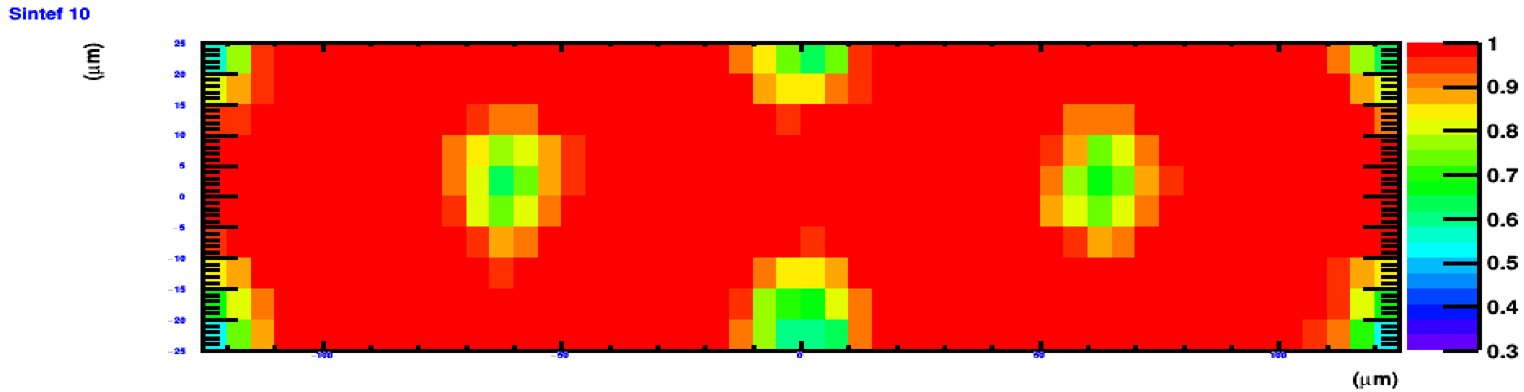
Hitmaps & efficiencies: First pass

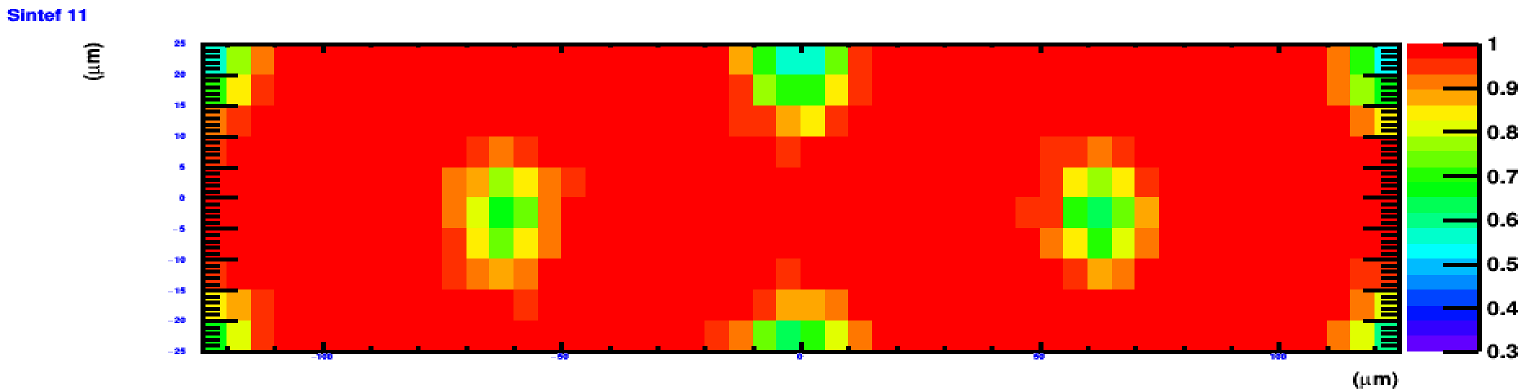


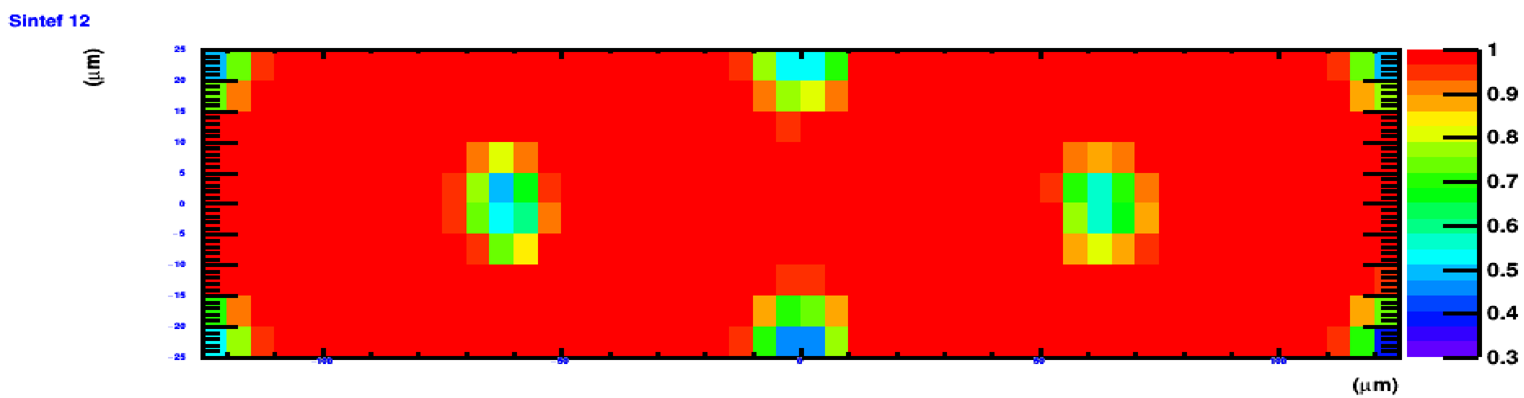
Individual pixel efficiencies.
Statistics per pixel is on the low side Averaging improves precisio
Ok *if we can remove pixels that are 'dead',*
for external reasons

Bump bonding problem: Remove by only considering event when track is predicted to hit a pixel with efficiency above 50%

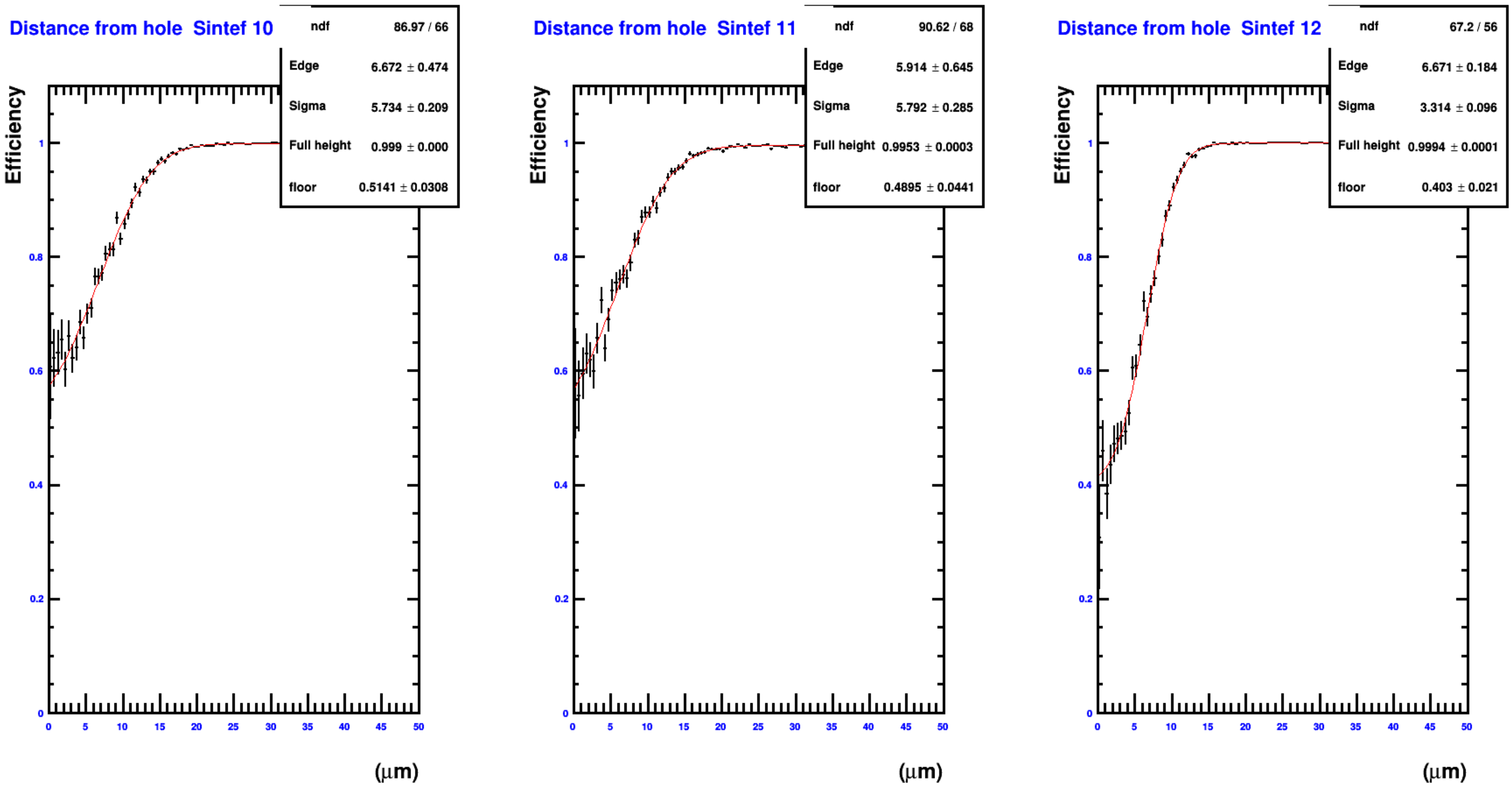
Averaged efficiency maps of a single pixel



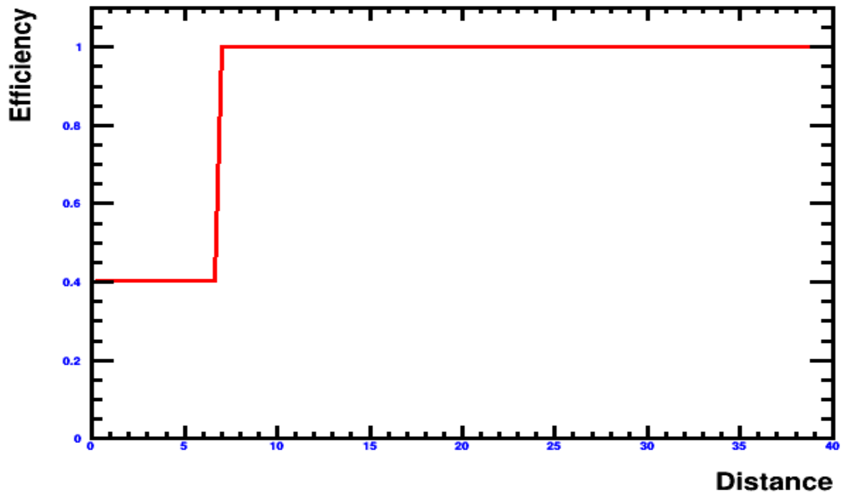


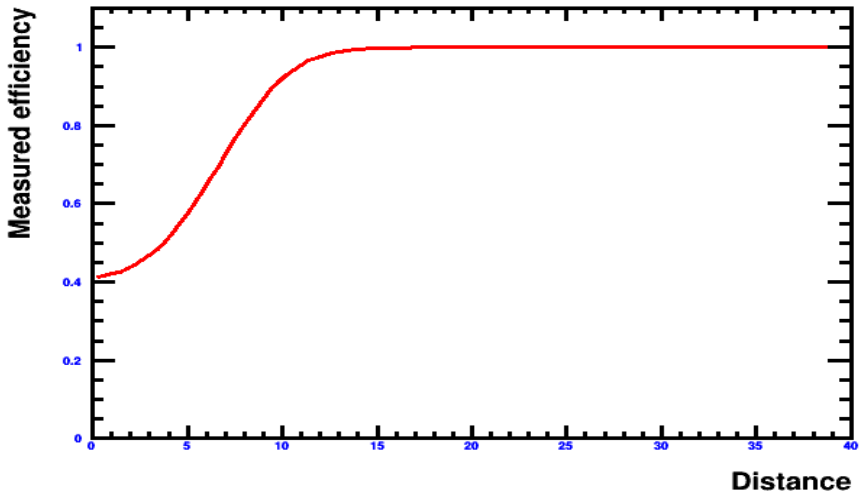


Average pixel efficiencies as function of predicted distance to nearest electrode ("hole")



Fit model: Step function: Two efficiencies as function of distance, *r,* from the hole center. Smeared with gaussian resolution function.
Parameters: Efficiencies inside and outside hole, the hole radius and the the resolution





Reproduces reasonably the hole diameters (12-14 microns), and impact resolution. Efficiency outside the hole is very close to 1. Efficiency inside the hole in the range (40-50)%.

'Theoretical' efficiency: 97.0%

Assuming hole diameter 14 um

40% efficiency in the holes

100% efficiency elsewhere

Measured efficiencies, for pixels that are 'alive' (efficiency > 50%).

Means should correspond to average pixel sensor efficiency