Status Update of DATCON

Belle II Tracking Meeting 06.04.2018

Christian Wessel, Bruno Deschamps,
Jochen Dingfelder, Carlos Marinas, Florian Bernlochner*

Universität Bonn, *KIT

wessel@physik.uni-bonn.de

Recap

- Until last update, tracking within the DATCON basf2 implementation was performed based on SVDTrueHit information (w/o background) or SVDSimHit information (w/ background)
- The performance was good and the results were promising, but input data were not what will be available for the FPGA implementation → needed to change this!
- Now: simplified clustering of SVD data with very simple noise filter and SpacePoint creation based on this
- Clusterizer and SpacePointCreator are adapted / simplified versions of SVDSimpleClusterizerModule and SVDSpacePointCreatorModule → should be easy to convert DATCONSVDCluster / DATCONSVDSpacePoints into SVDCluster / SpacePoint for comparison / debugging / RecoTrack creation
- Still to do: perform all SpacePoint creations with ‘custom’ algorithm
  - Right now I’m using the framework functions to convert local <-> global or to get the correct coordinates for a cellID
  - On FPGA everything has to be implemented, so I have to do this in basf2 as well
    - SVD mapping (FADC, APV, APV-channel…)
    - Conversion local (hit) coordinates into global coordinates (for HoughTrafo)

Simple Noise Filter

Idea of the noise filter: assume APV signal as t/tau*exp(-t/tau) and derive conditions

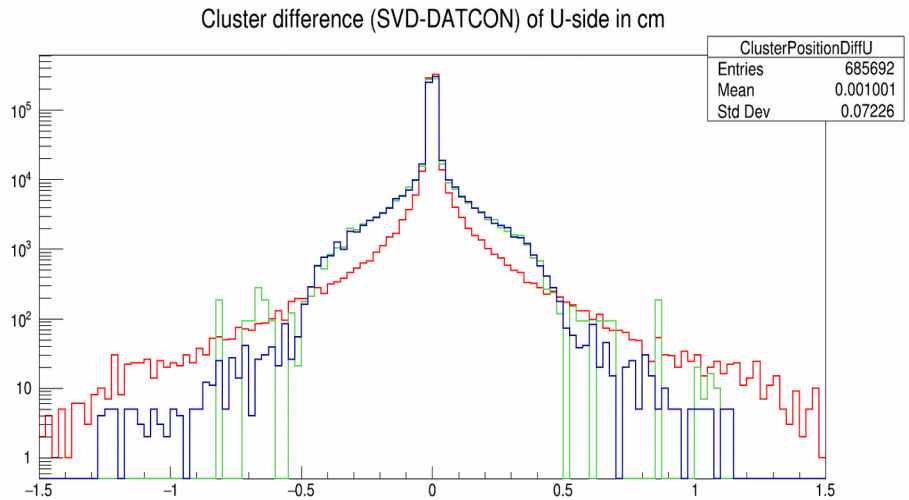
```
bool DATCONSVDSimpleClusterizerModule::noiseFilter(DATCONSVDDigit datconsvddigit)
{
  bool passNoiseFilter = false;  /**< initialize the passNoiseFilter with false as default */
  unsigned short maxSampleIndex = datconsvddigit.getMaxSampleIndex();
  DATCONSVDDigit::APVRawSamples sample = datconsvddigit.getRawSamples();

/* the initial idea was to declare a strip signal as noise if the samples neighbouring the maximum samle
 * (i.e. maxSampleIndex-1 and maxSampleIndex+1) had a charge (in ADU) of less then getMaxSampleCharge / 2 */
//   unsigned short noiseCut = sample[maxSampleIndex] / 2;

  // for the results shown m_NoiseCutInADU was set to 3
  unsigned short noiseCut = m_NoiseCutInADU;

  if (sample[maxSampleIndex] <= m_NoiseLevelInADU /*|| noiseCut < m_NoiseLevelInADU*/) {
    return false;
  }

  /* The maximum sample is expected to be 2nd or 3rd, so maybe change the conditions
   * in the if-clause accordingly to maxSampleIndex > 1 && maxSampleIndex < 4
   */
  if (maxSampleIndex > 0 && maxSampleIndex < (sample.size() - 1)) {
    if (sample[maxSampleIndex - 1] >= noiseCut && sample[maxSampleIndex + 1] >= noiseCut) {
      passNoiseFilter = true;
    }
  }

  return passNoiseFilter;
}
```

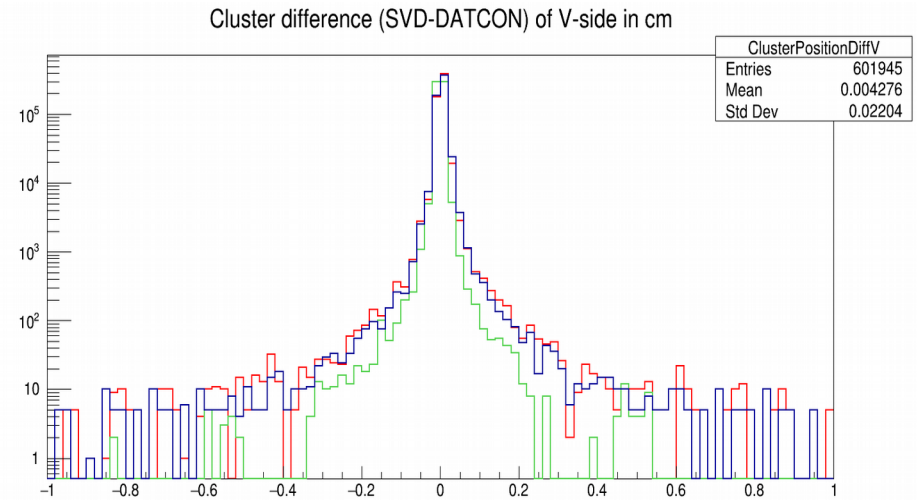
Simulation setup

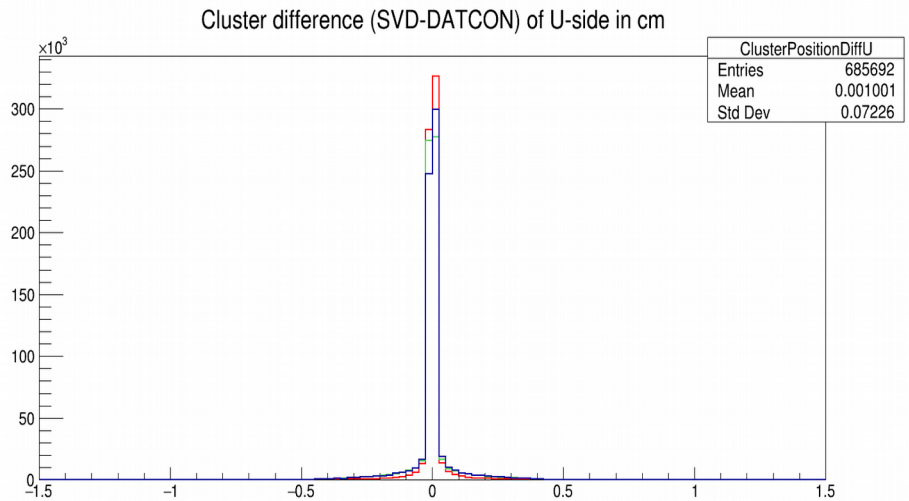
- 5000 Y(4S) events without background
- Simulations with background to follow
- NoiseCutInADU in simple noise filter was set to 3 (this value is not optimised)
- Self-made analysis function for clusters and SpacePoints (can be found in my branch for BII-3106)
  - I tried to do some kind of truth matching, probably I didn't do it the best way possible, but it works and the results look reasonable
  - (Maybe) strange thing: DATCONSVDCluster and DATCONSVDSpacePoints look better in terms of residuals with SVDTrueHits compared to SVDSpacePoints (see next slides)
- Parameters used in the tracking algorithm only were optimised for old version relying on SVDTrueHits / SVDSimHits
  - Optimisation for custom clusters and SpacePoints has to be done in future

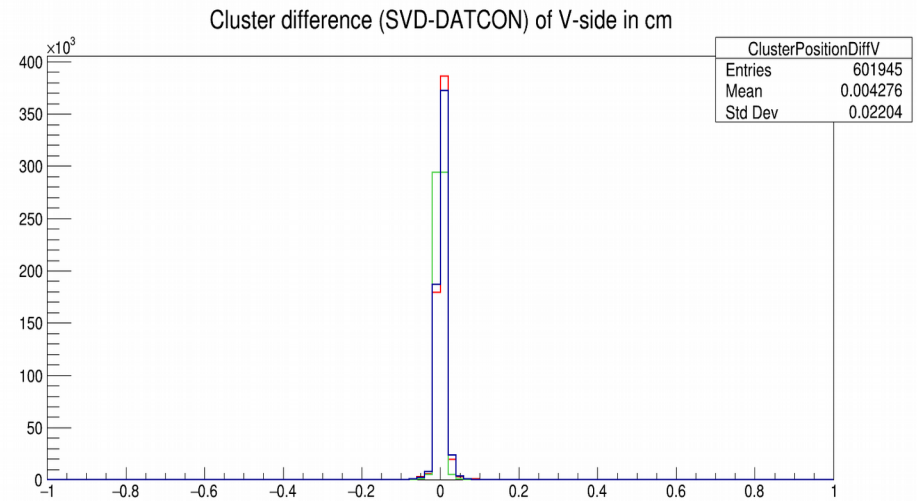
Simulation Results - Clusters

red = (SVDCluster-DATCONCluster), green = (SVDTrueHit-SVDCluster), blue = (SVDTrueHit-DATCONCluster)



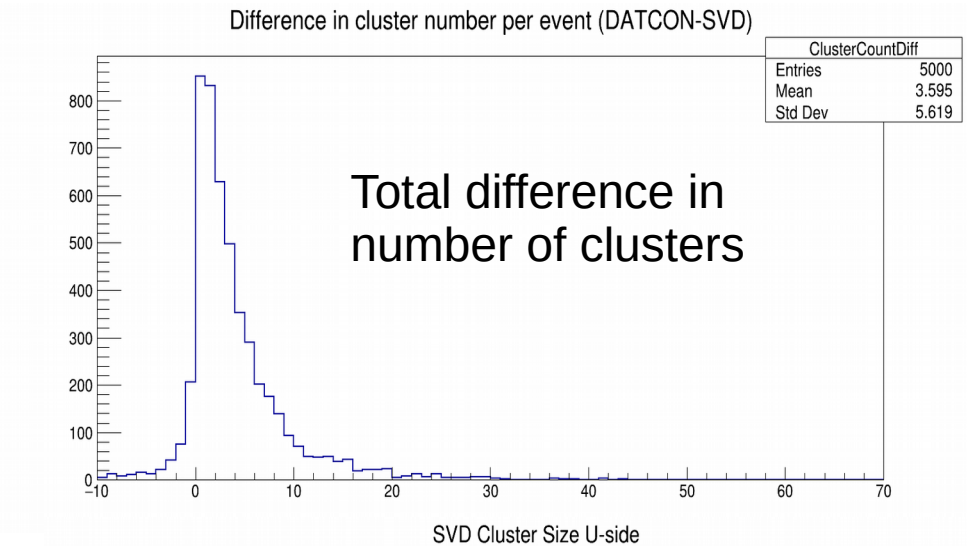


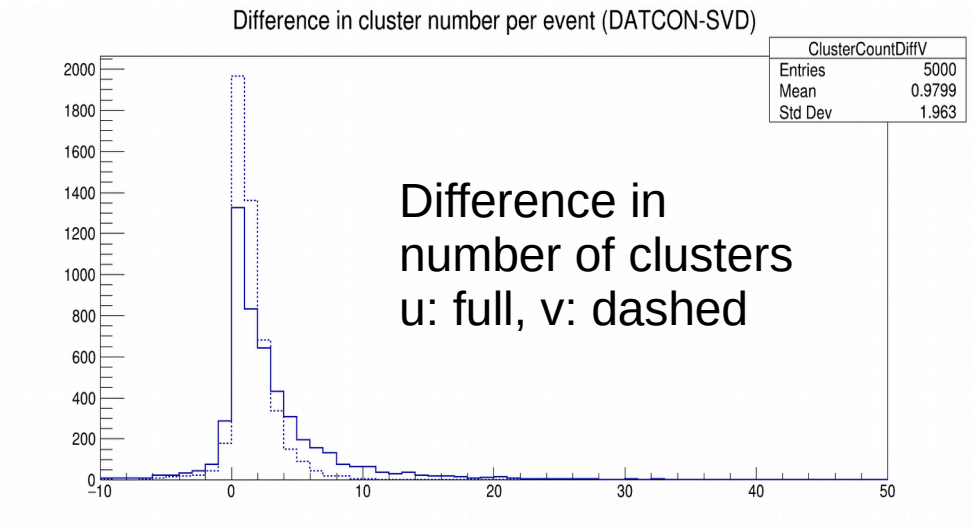


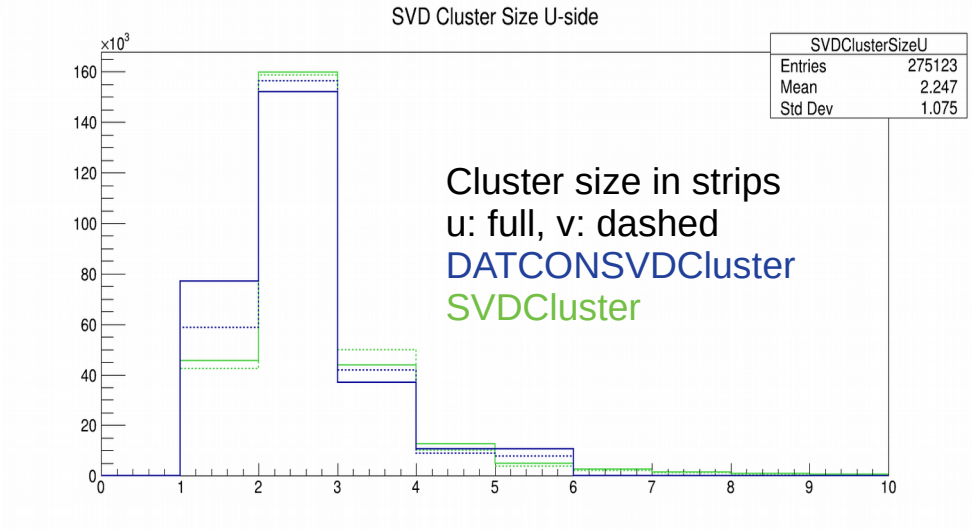


Simulation Results - Clusters

Difference in number of clusters and cluster size





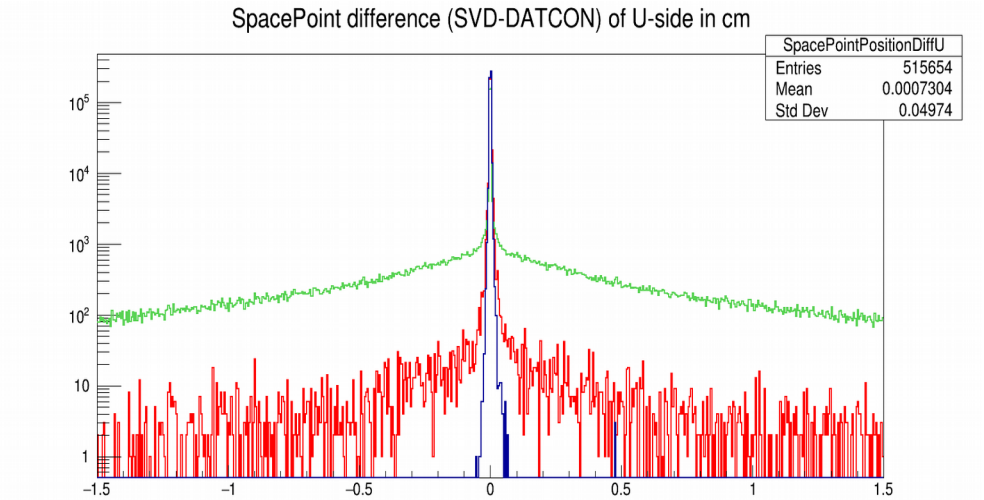


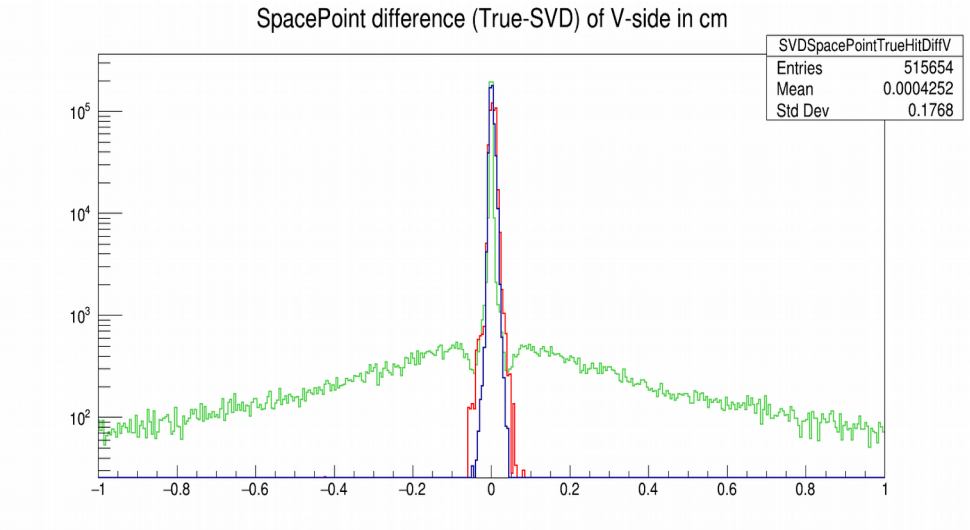
- Usually more DATCONSVDClusters than SVDClusters
- Strange cases with less DATCONSVDClusters than SVDClusters (unexpected)

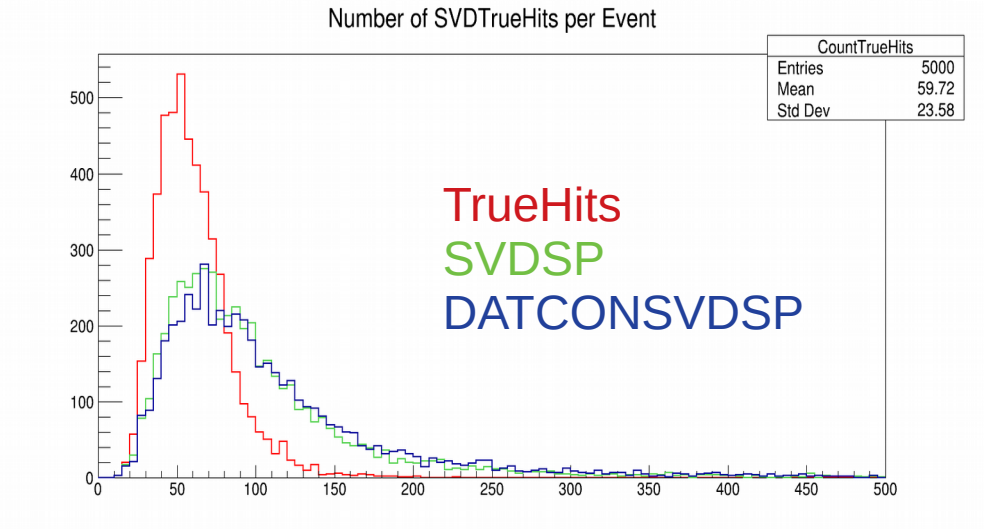
Simulation Results - SpacePoints

Difference of SpacePoint positions (u and v) and number of SpacePoints / TrueHits per Event

red = (SVDSP-DATCONSVDSP), green = (SVDTrueHit-SVDSP), blue = (SVDTrueHit-DATCONSVDSP)

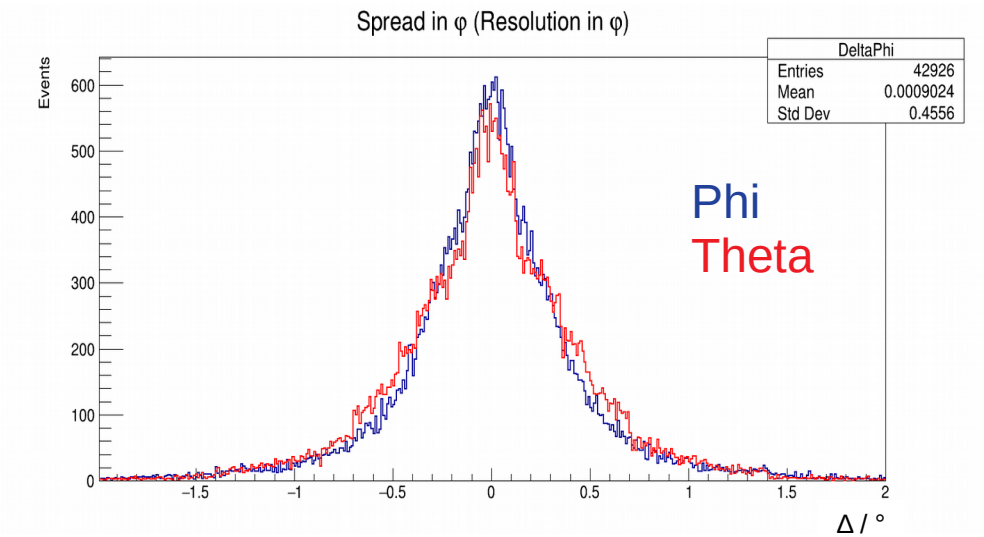


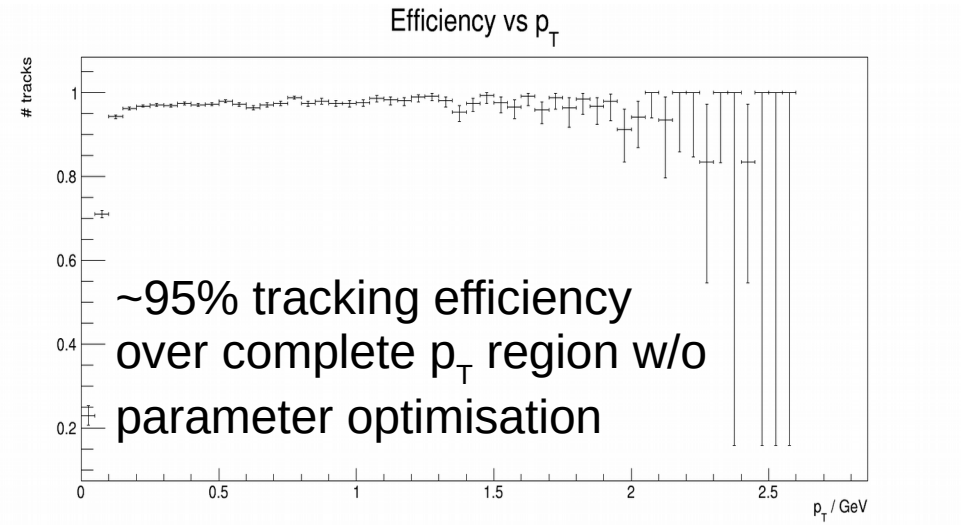




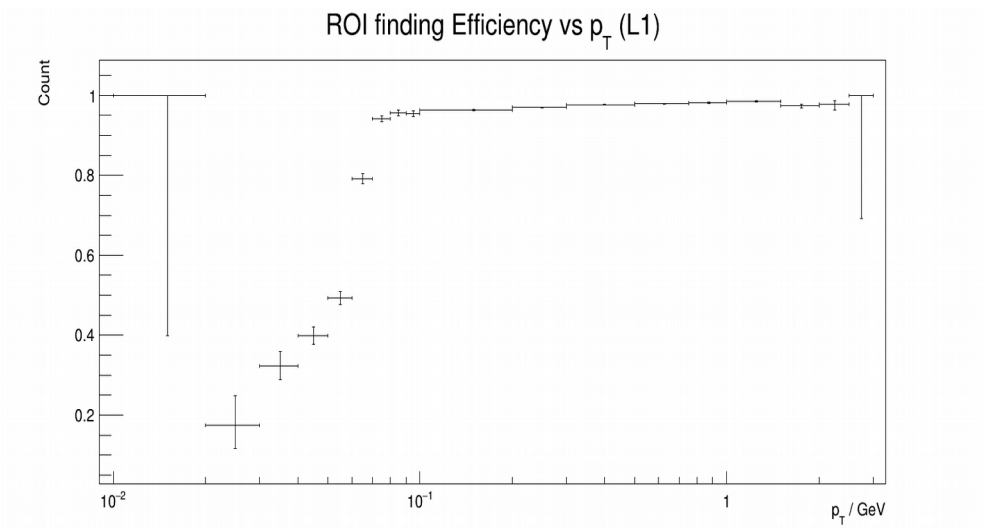
- Number of DATCONSVDSpacePoints and SVDSpaceSpoints almost the same
- Per event more SpacePoints than SVDTrueHits

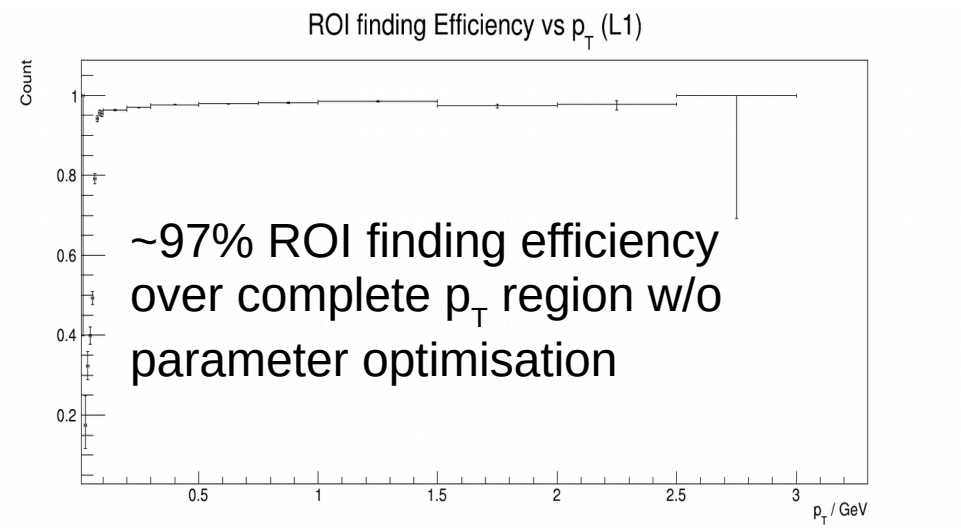
Simulation Results – Tracking & ROI





~95% tracking efficiency over complete p_T region w/o parameter optimisation





~97% ROI finding efficiency over complete p_T region w/o parameter optimisation

Summary & Conclusions

- Very good performance of finding SVD clusters even with simplified methods for clustering and noise filtering and having no timing information
- Very good performance of SpacePoint estimation
- But: My “truth matching” might be wrong, the residuals look “too good”
  - Still: the results are fine
- Even with the new input for the Hough Trafo (DATCONSVDSpacePoints) the tracking efficiency is ~95% and the ROI finding efficiency is ~97% (without parameter optimisation)
- Simulations with real SVDSpacePoints show very similar performance for tracking and ROI creation → DATCONSVDClusters and DATCONSVDSpacePoints seem to really be compatible :)

Outlook

- Simulations with background and comparison
- Optimisation of parameters!
- Still to be implemented and tested: calculation of local and global coordinates of clusters and SpacePoints without framework methods
  - Recent results look quite promising, no big performance decrease expected
  - Should only be some geometrical problems
- Save ROI with global classes implemented by Giulia → already in contact with Björn
  - Use global functions for performance estimation
- Save tracks as RecoTracks
  - Do performance studies for these RecoTracks (but ROI performance is more important)
- Clean up code and do documentation
- Remove / replace existing analysis functions
- ……..

UNIVERSITÄT BONN

SI LAB
Silizium Labor Bonn

Thank You!

bmb+f - Förderschwerpunkt
Elementarteilchenphysik
Großgeräte der physikalischen Grundlagenforschung

wessel@physik.uni-bonn.de