AGH University of Science and Technology

A modular software framework for test-beam data analysis. The **TbGaudi** package.

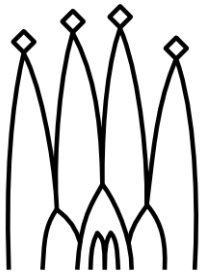
15 January 2019
Software tools

Bartłomiej Rachwał, Tomasz Szumlak
on behalf of the LHCb collaboration

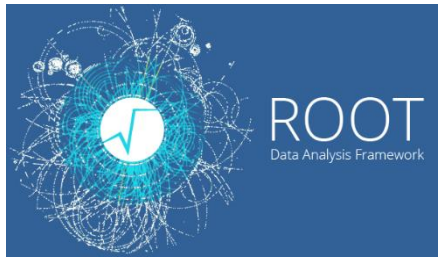
https://gitlab.cern.ch/TbAnalysis/Tb

Motivation | yet another framework?

- We would like to propose an universal design pattern – **application server type typical for the test beam experiments**.
- Existing frameworks such as **Gaudi are too heavy**, and **doing test beam data analysis standalone is not enough**!



GAUDI
Building HEP Data Processing Applications



CORRYVRECKAN

Common analysis framework (?)
algorithms and methods to be used by analysers in one place

Motivation | physics and prototype sensors

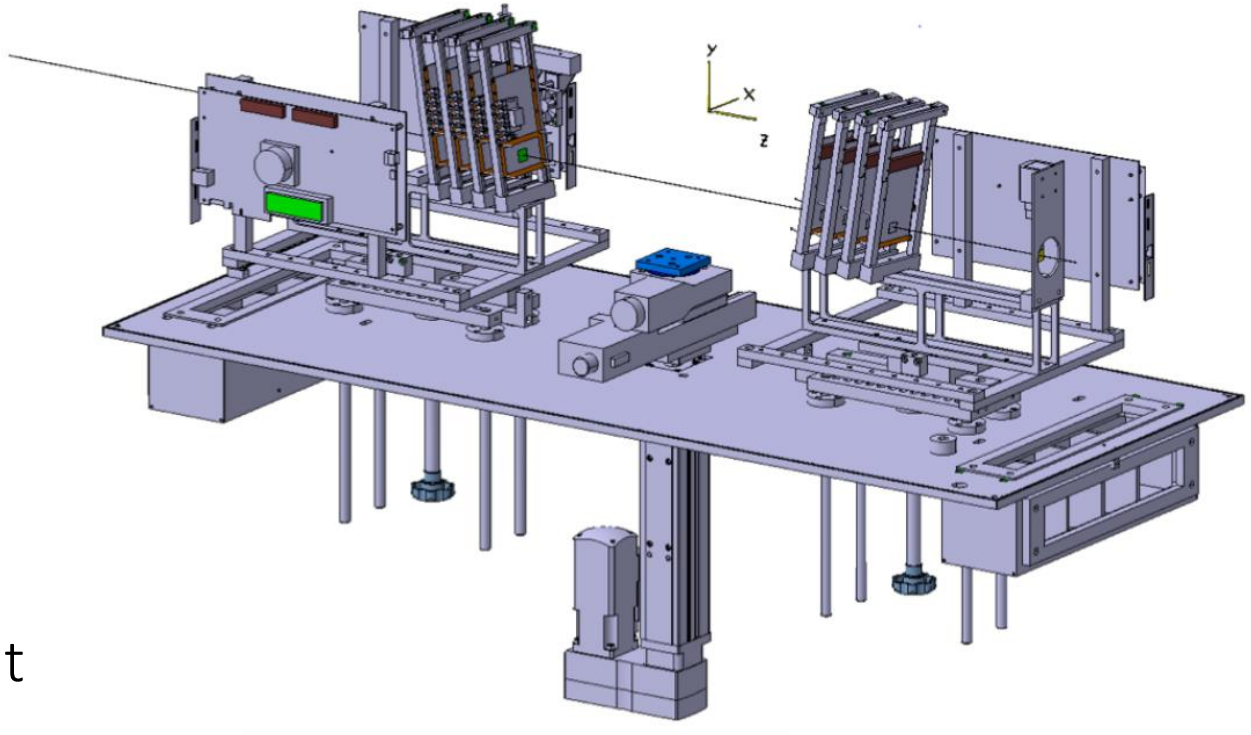
- **Understanding the impact of radiation damage on the detector performance:**
  - ✓ Charge collection efficiency
  - ✓ Position resolution single hit
  - ✓ Single hit reconstruction efficiency

- **Sensor configuration**
  - ✓ Sensor types (thicknesses, doping profiles, implant widths)
  - ✓ Inter-pixel isolation
  - ✓ Edge designs

- **Detailed studies with particle beam are essential for proper evaluation of detector prototypes**
  - ✓ After characterisation and calibration in the lab, the sensors need to be tested using high intensity particle fields, that are able to provide quickly high fluences up to 10^{16} [1 MeV n_{eq}/cm2]
  - ✓ <u>Uniform and non-uniform irradiation schemes</u>

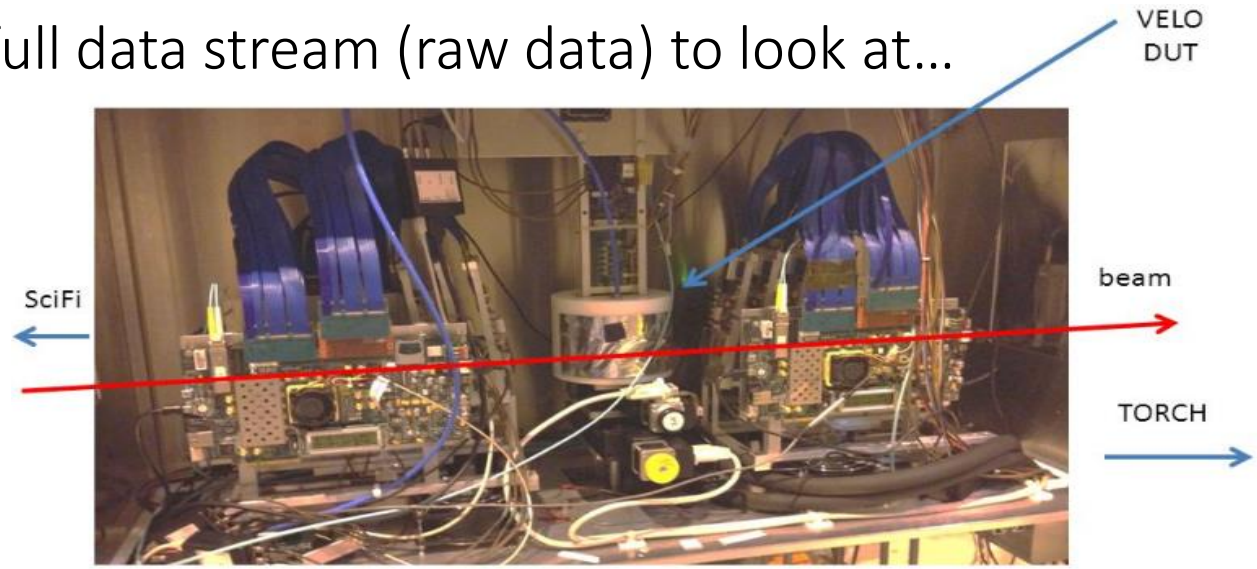
Motivation | physics and prototype sensors

- A dedicated high resolution pixel telescopes are being developed at different sites:
  - ✓ 180 GeV beam of protons and pions provided by the **SPS at CERN**
  - ✓ **Timepix3 telescope** - reconstructs the position and time information for the particles traversing the pixel sensor
- Our framework would be an ideal complement for these testing ecosystems!



- **The data produced by the telescope can be synchronised with the information from the DUT (device under test)**

  - ✓ The LHCb software embeded in the Kepler project performs the decoding of raw data and produces compressed data stream (tracks and clusters) objects inside the official LHCb's Gaudi framework.
  - ✓ One can consider full data stream (raw data) to look at…



  - ✓ What next?

Motivation | physics and prototype sensors

- **The data produced by the telescope can be synchronised with the information from the DUT (device under test)**

  - ✓ The LHCb software embeded in the Kepler project ne
    and produces compressed data stream (trac
    LHCb's Gaudi framework.
    official
  - ✓ One can consider full data

Prototyping and testing of new materials and sensor designs is cumbersome and time consuming. Having a comprehensive software platform to handle various test beam data sets is essential for efficient data analysis.

  - ✓ What next?

Tb Analysis | General workflow

Data

Experiment processed

⇒ Ntuples

(e.g. Clusters & Tracks)

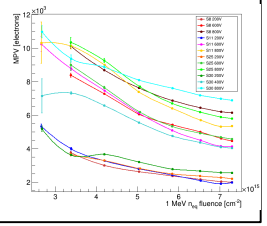
⇒ ASCII files

(e.g. ECS frames)

⇒ Raw data?

s/w

Plots



- Different data type for Tb analyses:
  e.g.: **.root files** prepared under exact configuration
- **Shared Tb data** to be analysed (e.g. EOS area)
- All Tb analysers **use same preprocessed data**

- Common analysis framework – algorithms and methods to be used by analysers in one place
- Common plotting style, fitting procedures etc. which can be easily adjusted if necessary

- Easy compilable and reproducible Tb results
- Series of control plots
- Systematics

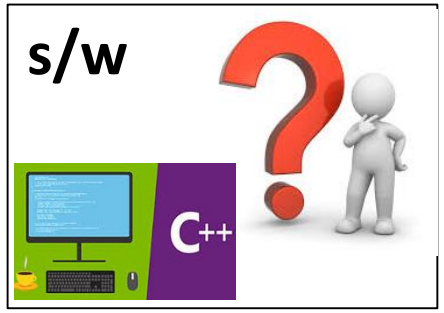
Pros and cons

+ Easy to combine series of Tb measurements
+ Repeat a selected analysis (e.g., new DUT, new data sample or just a cross-check)
+ Different analyses added as plugin tools, e.g. elliptical binning (motivated by different irradiation schemes)
+ Produce plots in the same style

- Two stages of data processing - need to pre-process data in test-beam facility s/w

The **TbGaudi** framework

- ✓ **General** for experiment specific purposes…
- ✓ **C++**, python, …
- ✓ Modular…
- ✓ User friendly…
- ✓ **Easy to adapt** by any group that is working on new sensor technology.



TbGaudi

- Data (Ntuples) Svc
- Binning schemes
- Irradiation schemes
- Fitting dataset handler
- TbAnalysis abstract interface
- …

TbExpAnalysis inherits from TbGaudi::TbAnalysis

- Data and Tb conditions
- Fitters definitions
- Irradiation scheme conditions (eg. beam allignment)
- Results combination

- ✓ **TbVeloPixAnalysis**
- ✓ **TbUTAnalysis**
- ✓ **Other requests ? ;)**

TbVeloPixAnalysis

```
// TbGaudi libriaries
#include "TbGaudi.h"
#include "TbStyle.h"
#include "TbAnalysis.h"
#include "TbRun.h"
#include "TbResults.h"

// Project libriaries
#include "TbVeloPixAnalysis.h"

int main(int argc, char **argv)
{
    TApplication* rootapp = new TApplication("App", &argc, argv);

    TbGaudi::Severity("Debug");

    auto MyAnalysis = std::make_unique<TbVeloPixAnalysis>();
    //MyAnalysis->Ana_CCE_IRRAD();
    MyAnalysis->Ana_ECS_General();

    rootapp->Run(!gROOT->IsBatch());
    return 0;
}
```

Inheritance from the TbGaudi library…

(0) Application (User's) main function

This is just an example processing chain, can make it simpler or more complex depending on the purpose! It is fully 'mix and match'!

main() → Ana() → TestBeamRunsLoop() → EventLoop() → DataFitting() → ResultsCompilation()

The TbGaudi framework | user analysis code

TbVeloPixAnalysis

```
5   ////////////////////////////////////////////////////////////////////////////////
6   ///
7   void TbVeloPixAnalysis::Ana_CCE_IRRAD()
8   {
9       // *** DEFINE DATA ***
10      TbAnalysis::TbDataDB(TbGaudi::ProjectLocation()+"/TbVeloPix/TbVeloPixAnalysis/data/TbDB_S8.dat");
11
12      // *** DEFINE ANALYSIS REQUIREMENTS ***
13      TbRun::GlobalClusterSize(2);
14      DutBinning::GlobalType("Elliptic");
15      DutBinning::GlobalNBins(4);
16
17      DutBinning::GlobalFluenceProfile("IRRAD");
18      DutBinning::FluenceMapping(true); // perform fluence mapping computation
19                                        // -> requires to register a dedicated fluence profile
20      // Event loop related data
21      DutRootUtils::WriteHistToNTuple(true);
22
23      // *** LOAD DATA ***
24      LoadTbData();
25
26      // *** RUN METHOD DEDICATED TO THIS ANALYSIS ***
27      CCE_IRRAD_MpvMapMaker();
28
29      // *** PLOT / COMBINE RESULTS ***
30      auto tbResults = TbResults::GetInstance();
31      tbResults->DrawDutMap("MPV");
32      tbResults->DrawDutProfile("MPV");
33      tbResults->DrawDutProfileCombined("MPV","column");
34      tbResults->DrawDutProfileCombined("MPV","row");
35      tbResults->DrawDutMap("Fluence");
36      tbResults->DrawCombined("MPV","Fluence");
37
38      //TbAnalysis::OutputLocation("anywhere..."); // by default it's set to project_location/TbResults
39      tbResults->Write(); // export everything to .pdf files
40
41
42  }
```

(1) **User defined** input data

(2) General analysis requirements

(3) **Automatic** data handler

(4) **User defined** test beam runs and event loops

(5) Results visualisation

(6) Write results to file

main() → **Ana()** → TestBeamRunsLoop() → EventLoop() → DataFitting() → ResultsCompilation()

(1) User defined input data

```
# Study of the different binning type and number of bins for the S8 device
# Author: Bartek Rachwal

<head>
<NTuple> ${DutName}_${RunBiasVoltage}V_${RunNumber}
</head>

<data> RunNumber,DutName,DutVendor,RunBiasVoltage[V],DutBinningType, DutNBins


<Path> /afs/cern.ch/user/b/brachwal/workspace/DATA/ntuples/Upgrade/testbeam/CCE/
12038,S8,HPK,600,Elliptic, 4
12038,S8,HPK,600,Uniform, 16
#12047,S8,HPK,300,Elliptic, 6
#12047,S8,HPK,300,Uniform, 16
</data>
```

← Template of the file name

← Number of columnes (analysis parameterization), as many as a user wants to define

← Files location

← test beam runs to be analysed

main() → **Ana()** → TestBeamRunsLoop() → EventLoop() → DataFitting() → ResultsCompilation()

(4) User defined runs and event loops method

Looks like typical routines dedicated to Ntuple loop, but ...

- Tools to DUT area binning (e.g. square, elliptic, strip)
- Tools to define histogram(s) of RooDataSet(s)
- Tools to write produced data into files

- After the event loop, the data fitting can be performed automatically for series of produced histograms (or RooDataSets)

* User defined fitter

Inheritance from the TbGaudi library…

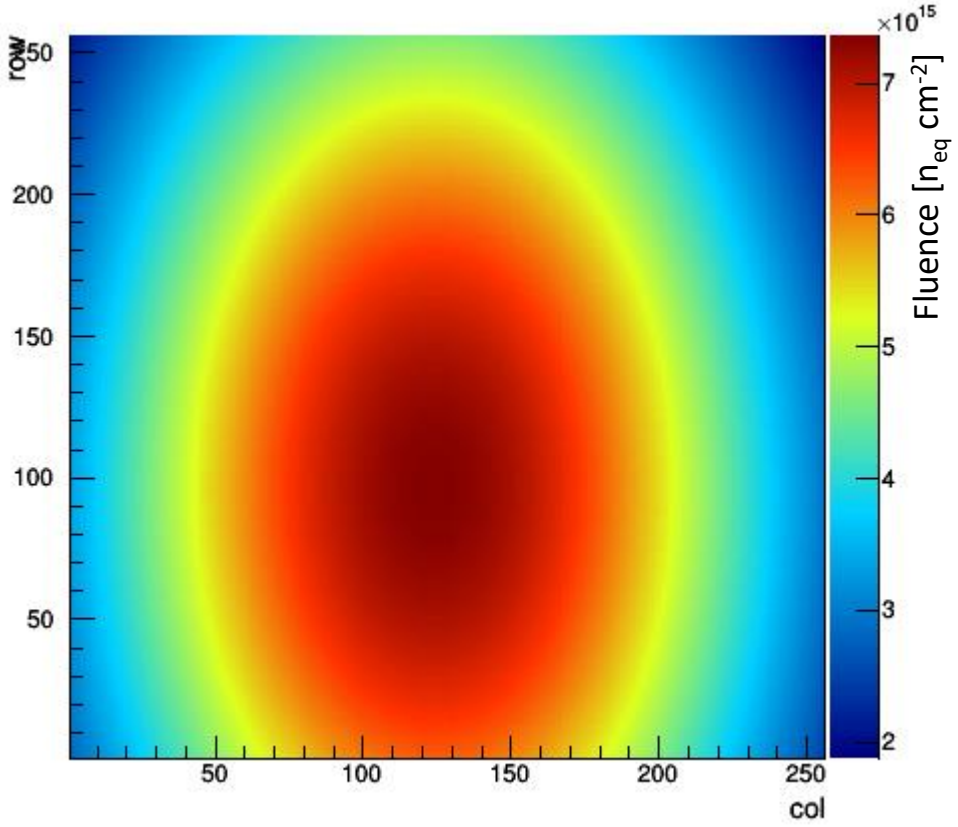
Dedicated class which has acces to the prepared data in the event loop (TH1 or RooDataSet). To start the fitting procedure, simply:

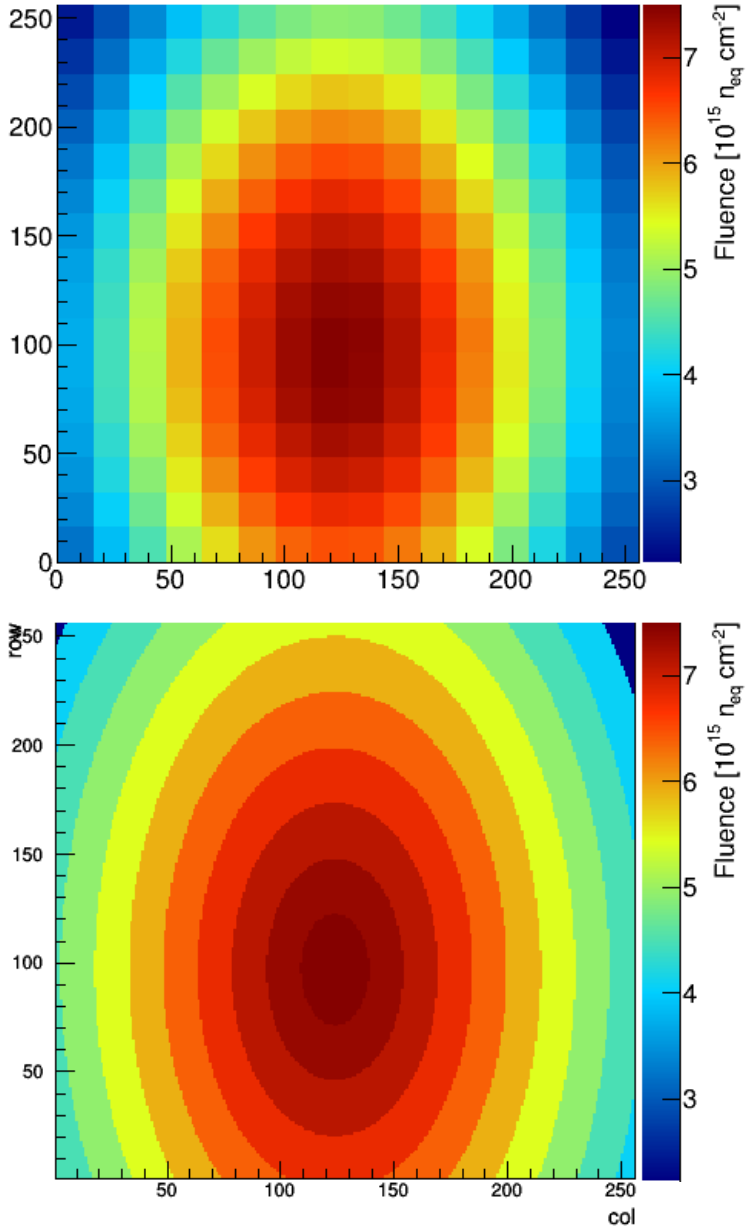
```
auto chargeFitter =  std::make_unique<ChargeFitter>();

chargeFitter->SetData(iTbRun, "Charge", FitDataType::RooDataSet);
chargeFitter->Roofit(); // run fitting

chargeFitter->SetData(iTbRun, "Charge", FitDataType::TH1);
chargeFitter->Root(); // run fitting

```

main() → Ana() → **TestBeamRunsLoop() → EventLoop() → DataFitting()** → ResultsCompilation()

Examples plots | TbVeloPixAnalysis

IRRAD* fluence profile maps





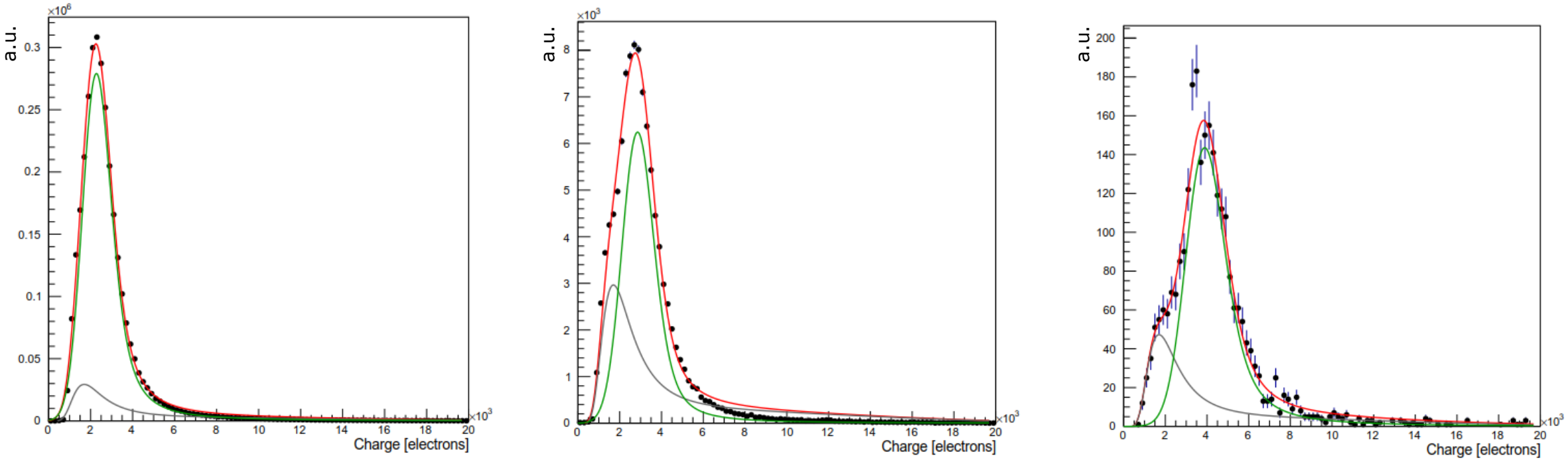
✓ Different radiation schemes are followed by adaptive binning – benefit for statistic and physics analysis (easier to interpret results)

* The IRRAD proton facility is located on the T8 beam-line at the CERN PS East Hall

Examples plots | TbVeloPixAnalysis

LHCb VeloPix Preliminary

Deposited energy fitting

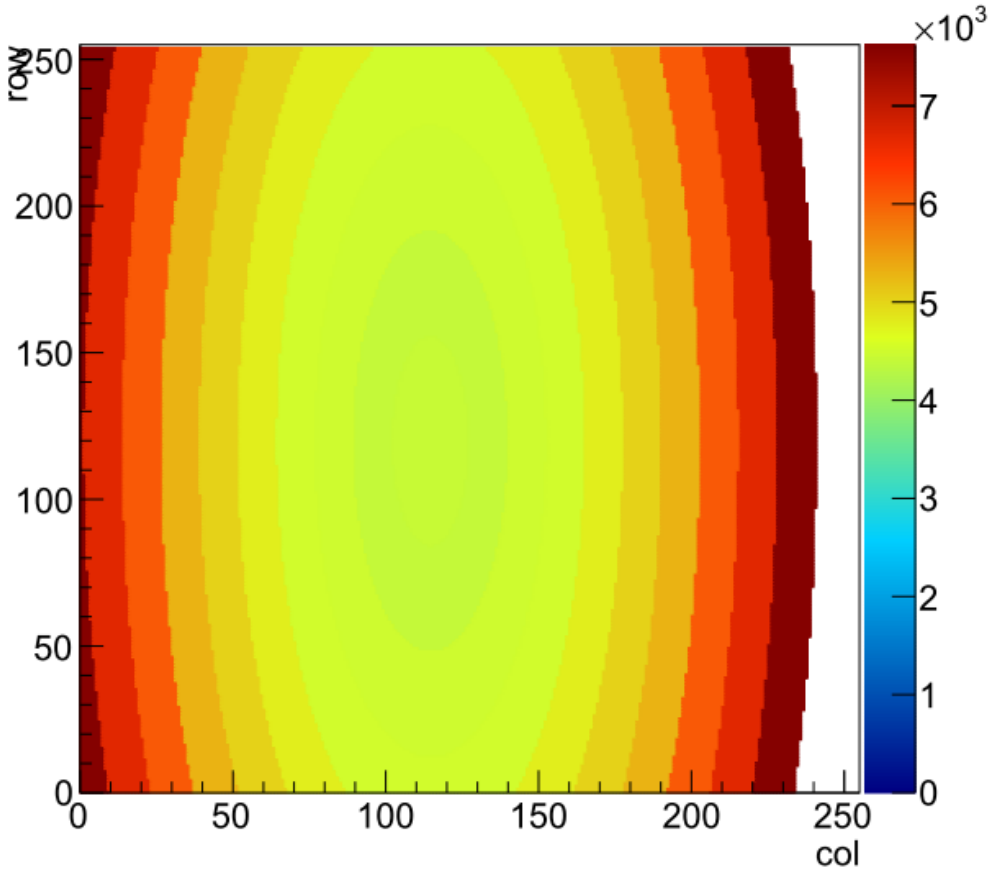


- ✓ Charge distribution is far from *simple* Landau curve description,
- ✓ Basic/generic fitters are defined within TbGaudi (Landau + Gauss convolution),
- ✓ However, the fitter can be extended by a given TbExpAnalysis implementation, e.g. taking into account background/noise modeling.

Examples plots | TbVeloPixAnalysis

LHCb VeloPix Preliminary

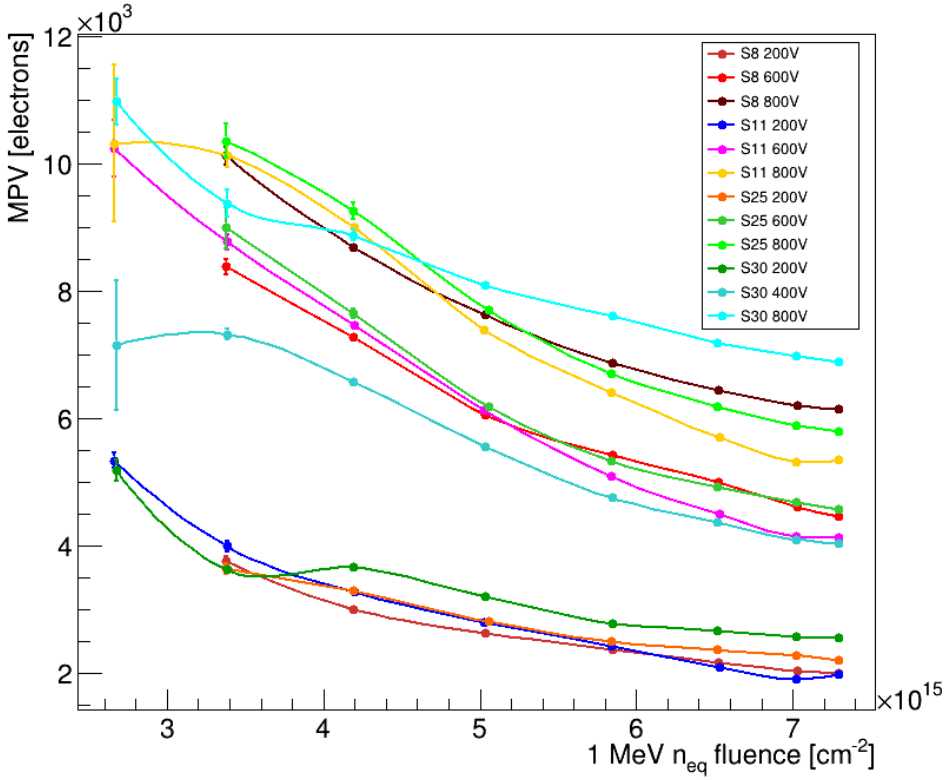
Most **P**robable **V**alue maps



Examples plots | TbVeloPixAnalysis

LHCb VeloPix Preliminary

Analysis curves



- ✓ Its really hard to **compile series of different DUT analyses** without a *framework*
- ✓ Its would be really easy to re-run the whole analysis – if needed.
  This is related to the **data preservation.**

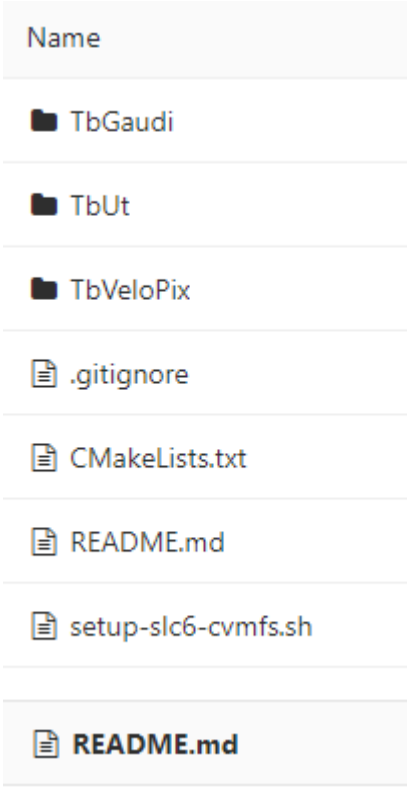
Summary

1. Series of test beam R&D studies are being performed by different groups/experiments;

2. By this presentation the TbGaudi (a modular software framework based mostly on ROOT) has been proposed for test-beam data analysis;

3. It's an easy to adapt framework by any group that is working on new sensor technology.

4. Currently, the framework and the analyzes packages are placed in same repository: **https://gitlab.cern.ch/TbAnalysis/Tb**



https://gitlab.cern.ch/TbAnalysis/Tb