05/17/2019

Detector simulation workflows on the GRID

Project 8

Sara Walcher & Lars Reining

supervisor: Sandro Wenzel

ALICE

WLCG – The Worldwide LHC Computing Grid

WLCG – The Worldwide LHC Computing Grid

- loosely coupled collection of heterogenous computing farms around the world
- world's largest computing grid
- 1 Exabyte storage
- Task: distributing, simulating, analysing, storing data from LHC
- easy access to GRID via GRID middleware as single entry point (AliEn)
- Crucial to CERN and LHC

Project description

Testing the functioning of ALICE software for Run 3 on the GRID

Not done before

↓

Testing detector simulation on the GRID

↓

Development of workflows to study influence of materials and parameters on detectors

↓

Specific analysis for compensator magnet and FD Detector

Part 1: Interacting with GRID

- Learned about ALICE GRID interaction
  - Loading ALICE Run 3 software Stack
  - File system on GRID
  - Job submission
- Basics of Bash-scripting and Linux
- How to deal with failure and practising patience

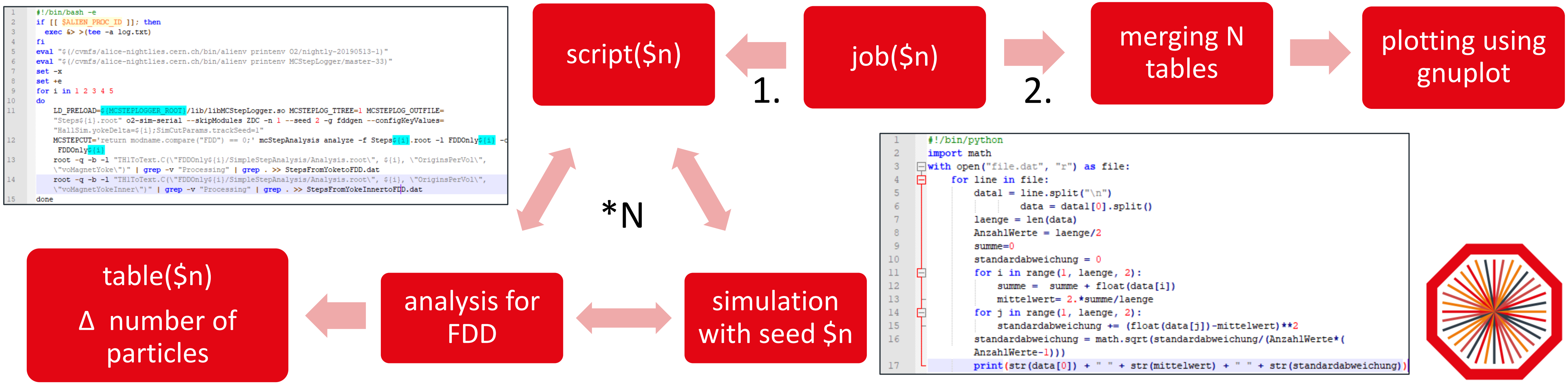
successful demonstration of correct execution of new detector simulation framework

```
Executable = "/alice/cern.ch/user/s/swenzel/hssip/YoketoFDD/FueberDeltaSkriptFDDGrid.sh";
Requirements = ( other.CE == "ALICE::CNAF::LCG" );
Arguments = "${1}";
InputFile = {
  "LF:/alice/cern.ch/user/s/swenzel/hssip/YoketoFDD/TH1ToText.C"
};
OutputDir = "/alice/cern.ch/user/s/swenzel/hssip/YoketoFDD/Output${1}";
Output = {
  "log.txt@disk=1"
};
```

```
#!/bin/bash
N=100
for n in {1..${N}}
do
    alien_submit FueberDeltaSkriptFDDGridJob.jdl ${n}
done
```

Part 2: ALICE detector simulation

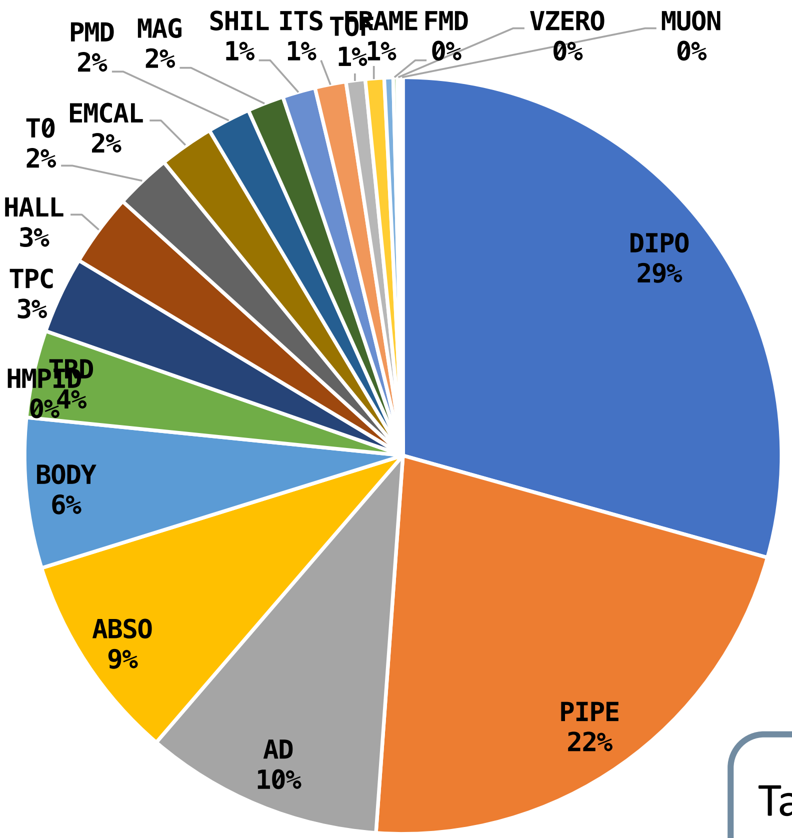
- Basics in C++ (Root) and Python to extract and analyse data
- Knowledge about the different detectors of ALICE
- How to automatize repeating tasks and to connect them into complex workflows




```bash
#!/bin/bash -e
if [[ $ALIEN_PROC_ID ]]; then
  exec &> >(tee -a log.txt)
fi
eval "$(/cvmfs/alice-nightlies.cern.ch/bin/alienv printenv O2/nightly-20190513-1)"
eval "$(/cvmfs/alice-nightlies.cern.ch/bin/alienv printenv MCStepLogger/master-33)"
set -x
set +e
for i in 1 2 3 4 5
do
    LD_PRELOAD=${MCSTEPLOGGER_ROOT}/lib/libMCStepLogger.so MCSTEPLOG_TTREE=1 MCSTEPLOG_OUTFILE=
    "Steps${i}.root" o2-sim-serial --skipModules ZDC -n 1 --seed 2 -g fddgen --configKeyValues=
    "HallSim.yokeDelta=${i};SimCutParams.trackSeed=1"
    MCSTEPCUT='return modname.compare("FDD") == 0;' mcStepAnalysis analyze -f Steps${i}.root -l FDDOnly${i} -c
     FDDOnly${i}
    root -q -b -l "TH1ToText.C(\"FDDOnly${i}/SimpleStepAnalysis/Analysis.root\", ${i}, \"OriginsPerVol\",
    \"voMagnetYoke\")" | grep -v "Processing" | grep . >> StepsFromYoketoFDD.dat
    root -q -b -l "TH1ToText.C(\"FDDOnly${i}/SimpleStepAnalysis/Analysis.root\", ${i}, \"OriginsPerVol\",
    \"voMagnetYokeInner\")" | grep -v "Processing" | grep . >> StepsFromYokeInnertoFDD.dat
done
```

```python
#!/bin/python
import math
with open("file.dat", "r") as file:
    for line in file:
        datal = line.split("\n")
                data = datal[0].split()
        laenge = len(data)
        AnzahlWerte = laenge/2
        summe=0
        standardabweichung = 0
        for i in range(1, laenge, 2):
            summe =  summe + float(data[i])
            mittelwert= 2.*summe/laenge
        for j in range(1, laenge, 2):
            standardabweichung += (float(data[j])-mittelwert)**2
        standardabweichung = math.sqrt(standardabweichung/(AnzahlWerte*(
        AnzahlWerte-1)))
        print(str(data[0]) + " " + str(mittelwert) + " " + str(standardabweichung))
```

Specific simulation question

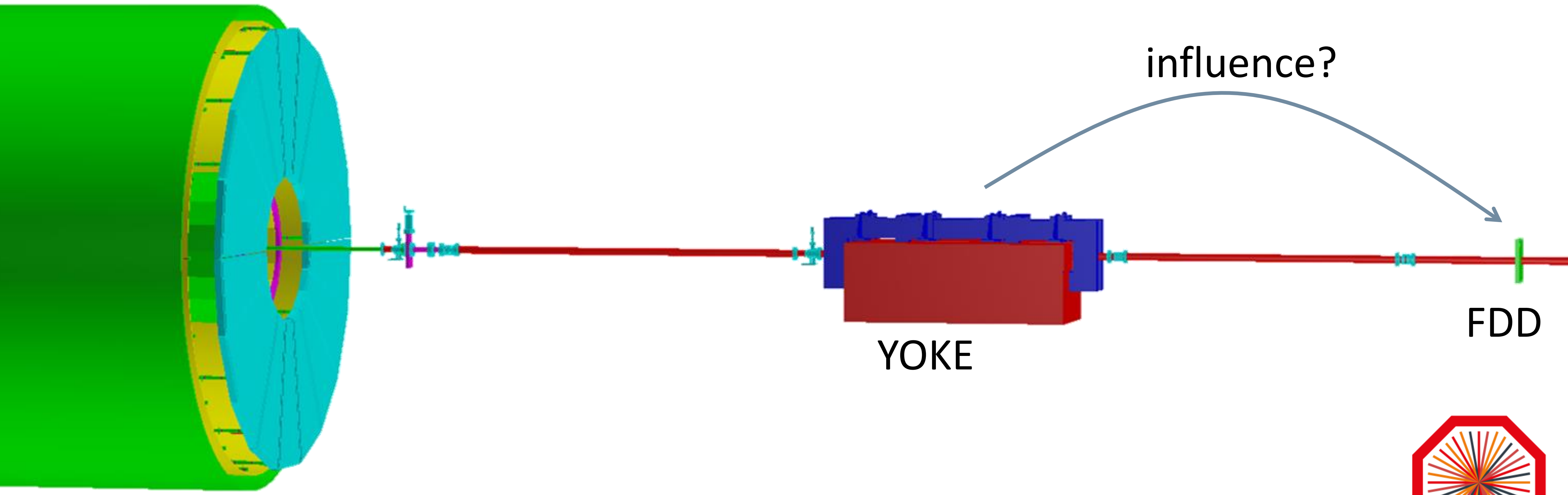


- Simulation may need lots of CPU time in certain materials
- Steps in DIPO make up 29% of simulated Steps
- DIPO mainly consists of YOKE in front of FDD

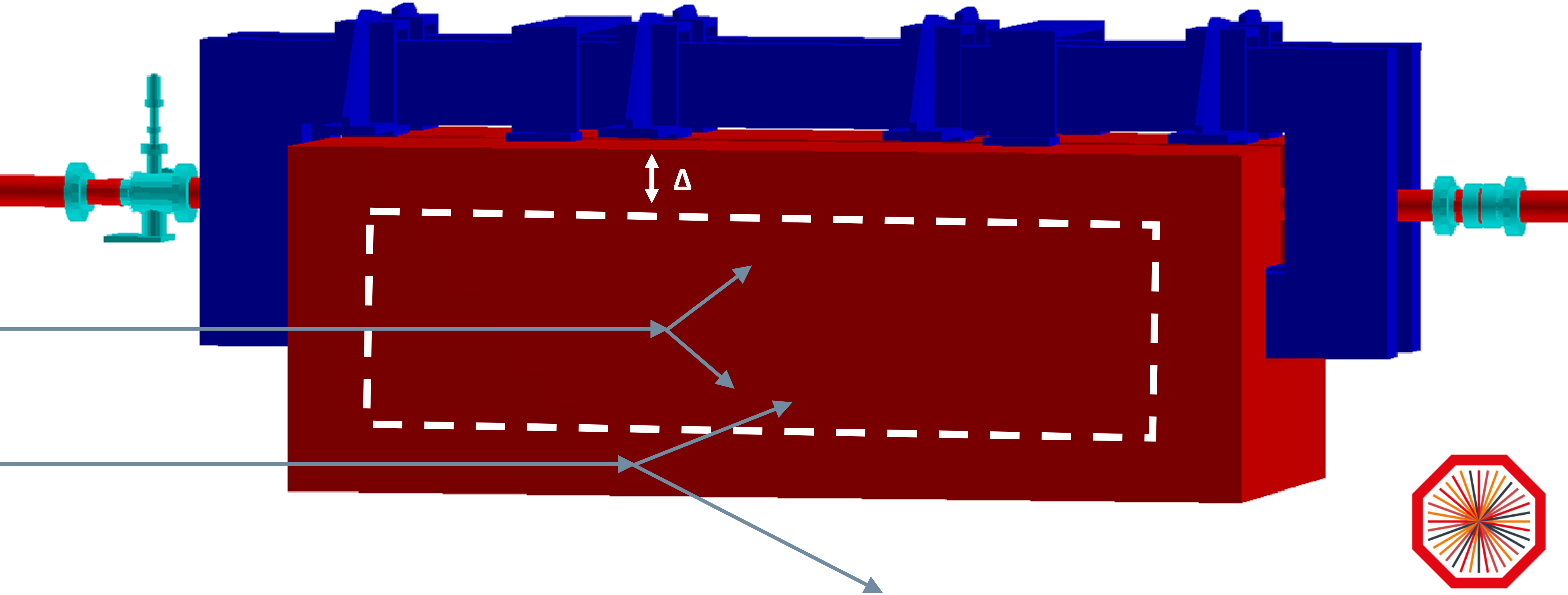
"How important are the secondaries created in YOKE?"

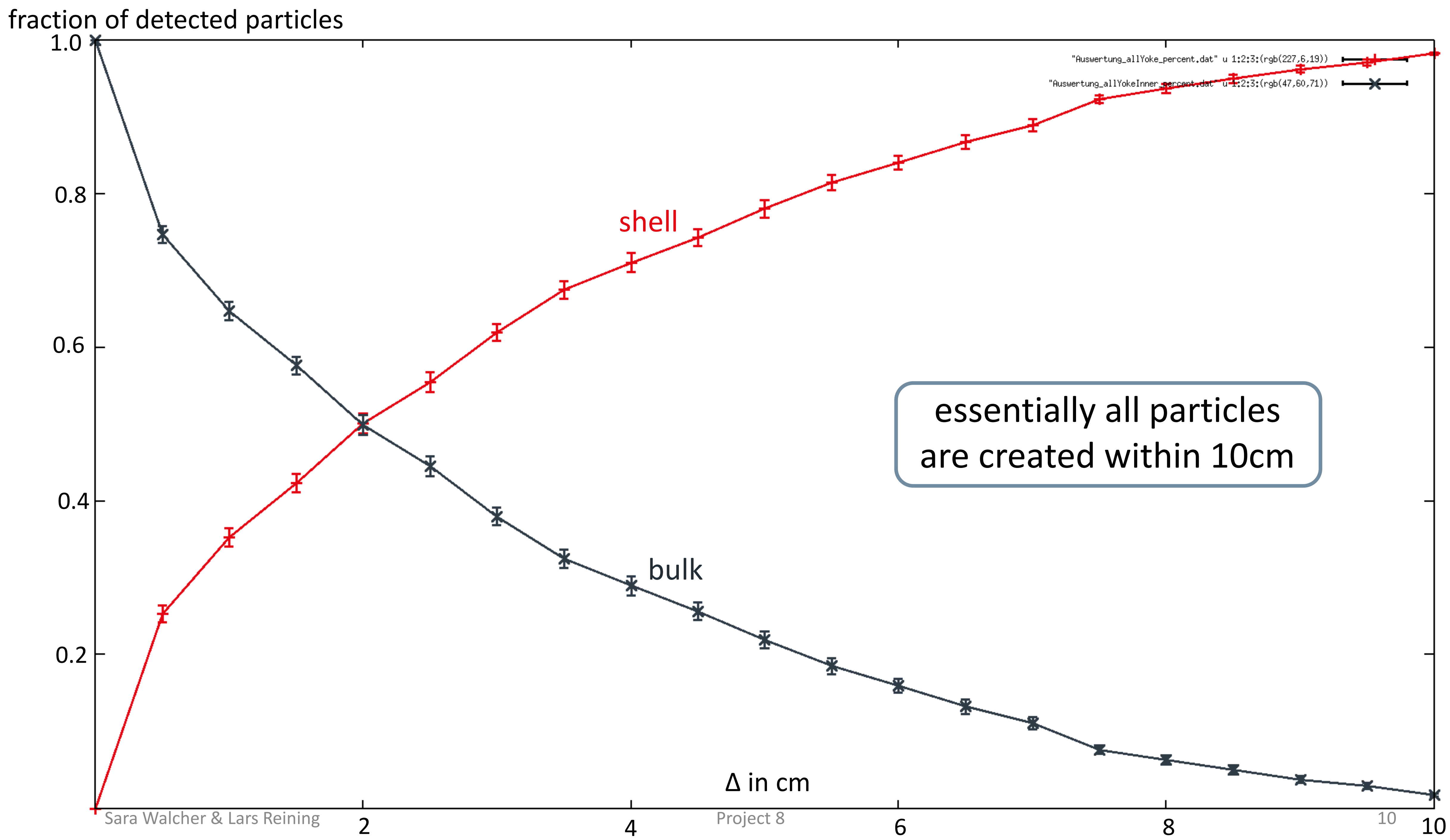
Targeting detailed analysis for where in YOKE secondaries are created for the purpose of material budget optimization

Visualizing the problem



Study: Fraction of secondaries seen by FDD as a function of Δ




fraction of detected particles
1.0
0.8
0.6
0.4
0.2
"Auswertung_allYoke_percent.dat" u 1:2:3:(rgb(227,6,19))
"Auswertung_allYokeInner_percent.dat" u 1:2:3:(rgb(47,60,71))
shell
bulk
essentially all particles are created within 10cm
Δ in cm
2
4
6
8
10

Thank You! Danke! Merci!

Project 8

Sara Walcher & Lars Reining

supervisor: Sandro Wenzel

ALICE