Experiment Simulation

Exercises

"The proof of the pudding is in the eating."

CSC 2004

Martin Liendl

Table of Contents

There I talk ....

what we are going to do

- which kind of simulation
- overview of the detector setup in Geant4
- primary generator
- event and hit definition, numbering scheme
- parameters and metadata
- analysis

exercises (mutual, different levels of difficulty)

... and here YOU work!

* complete the geometry
* implement the primary generator action
* implement the sensitive detector action
* implement a numbering scheme
* implement the analysis (if you already know G4)
* simply run several simulation jobs with different parameters (if you don't know, what's this all about)
* extend the simulation (other particles than e-, more hit information, ..) and modify the analysis accordingly
* extend the simulation for simple pile-up treatment and digitization

Table of Contents

Red Pill or **Blue Pill** ?

exercises (mutual, different levels of difficulty)

solutions provided

* complete the geometry
* implement the primary generator action
* implement the sensitive detector action
* implement a numbering scheme
* implement the analysis (if you already know G4)
* simply run several simulation jobs with different parameters (if you don't feel familiar with G4 at all)

solutions **not** provided

* extend the simulation (other particles than e-, more hit information, ..) and modify the analysis accordingly
* extend the simulation for simple pile-up treatment and digitization

What we are going to do:

Determine the **energy resolution of a crystal electromagnetic calorimeter** for electrons over a range of energies

- Set-up the simulation
  - define a crystal calorimeter
  - specify a primary particle generator shooting e- at random

today

- Run lots of simulations, store simulated hits & MC-truth

run over WE

- Analyze the persistent data
  . loop over persistent hits
  . sort and condense the simulated data by looking at MC-truth, compare with MC-truth

Monday

For simplicity:

- no pile-up
- no digitization

But we can "pretend" to have still a quite "realistic" analysis:

- simulated hits ~ measured signal
- MC-truth ~ information delivered by measurements of other sub-detectors (e.g. tracker information)

At the end of the day (of today!) ...

... in **one way** or **the other**, you should get acquainted with the basics of Geant4 during the exercises.

~15min before the end of the today's exercises:

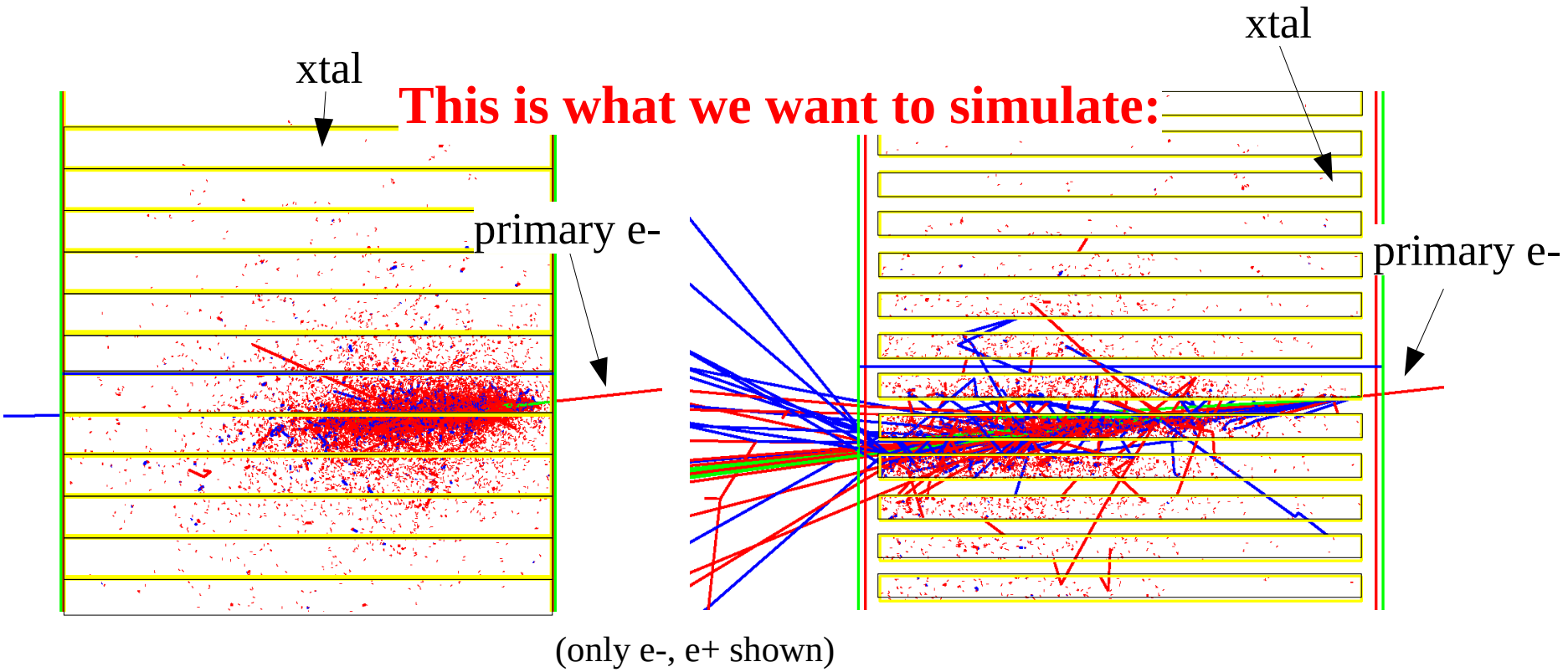
We will go into “massive data production”-mode!!
Each group will be assigned a different set of simulation parameters and start a simulation of several hundreds (or thousands?) events. These events will be stored locally using POOL. We will run the “solution” executable with POOL storage enabled.

On Monday:

On Monday we will combine all your results and analyze all the data (looping over POOL stored events, calculating AIDA histograms ...);

There will be time for your questions!!! (... and, hopefully, time for answers, as well ...)

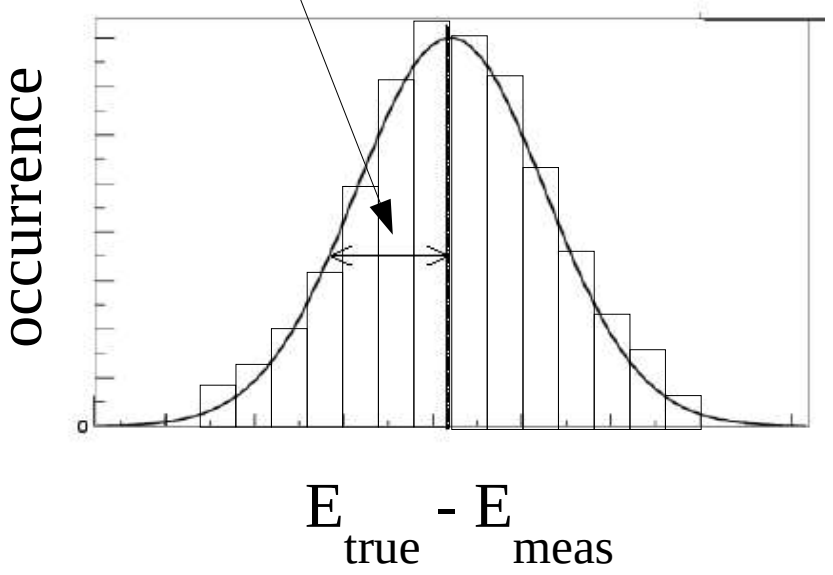
All exercises are about the same setup: a crystal calorimeter.
Every exercise has a different thing missing -> you have to complete it!



The geometry is parameterized with 3 free parameters. They influence the simulation result as show above: two showers with same incident energy, but different inter-crystal gaps.

What for?

$$\frac{\sigma_E}{E} = \frac{a}{\sqrt{E}} \oplus \frac{b}{E} \oplus c$$



$E_{true} - E_{meas}$

binned for many measurements

a ... due to the quantum character of an electromagnetic shower

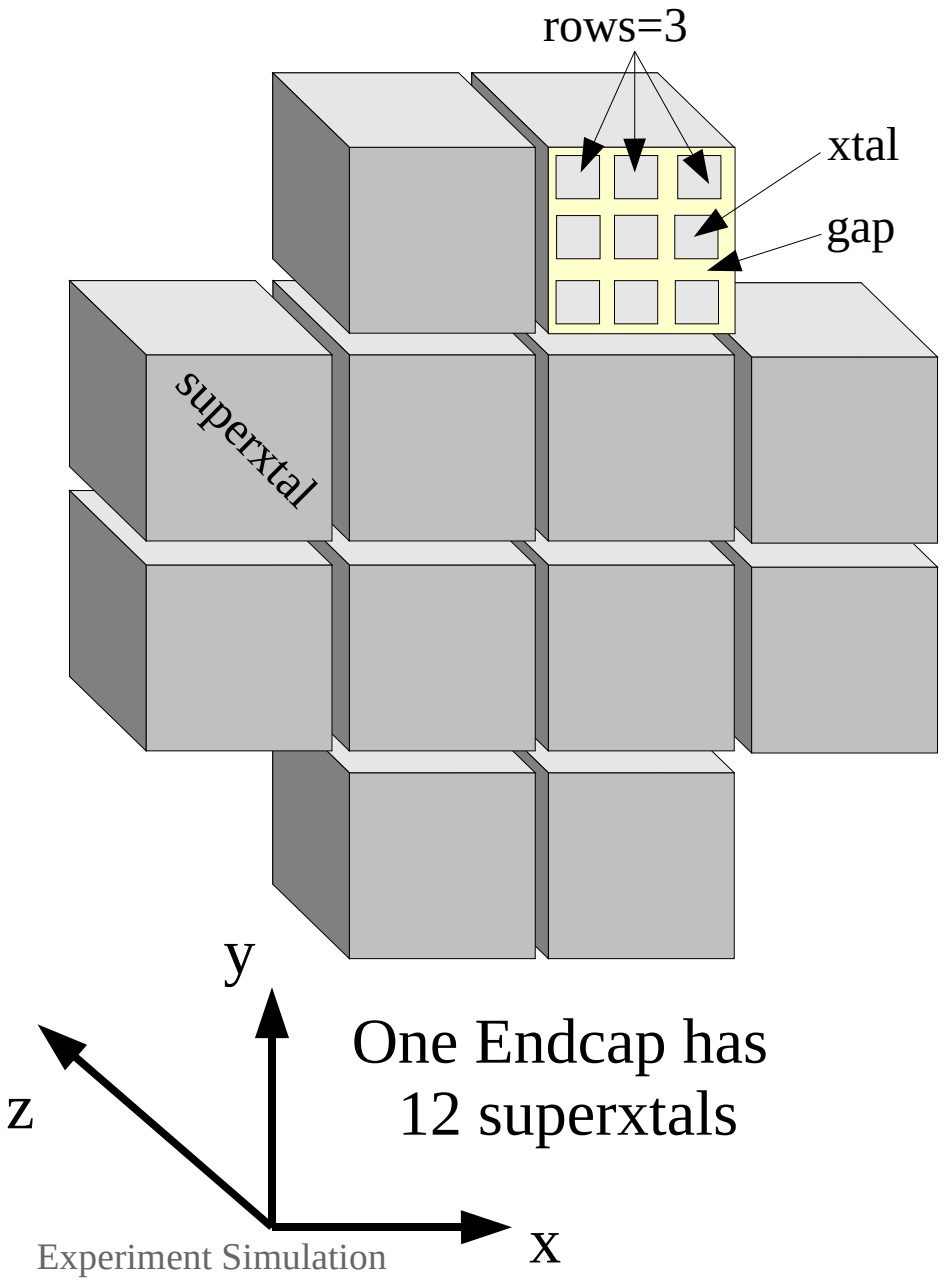
b, *c* .. due to other instrumental effects

One can estimate *a*,*b*,*c* through experiments where E_{true} is known.

One can estimate *a* through MC simulation!

If one finds agreement between several experiments and simulations, simulation can be "trusted" and used to study variations of setups without having to test it in experiment ...

Geometry

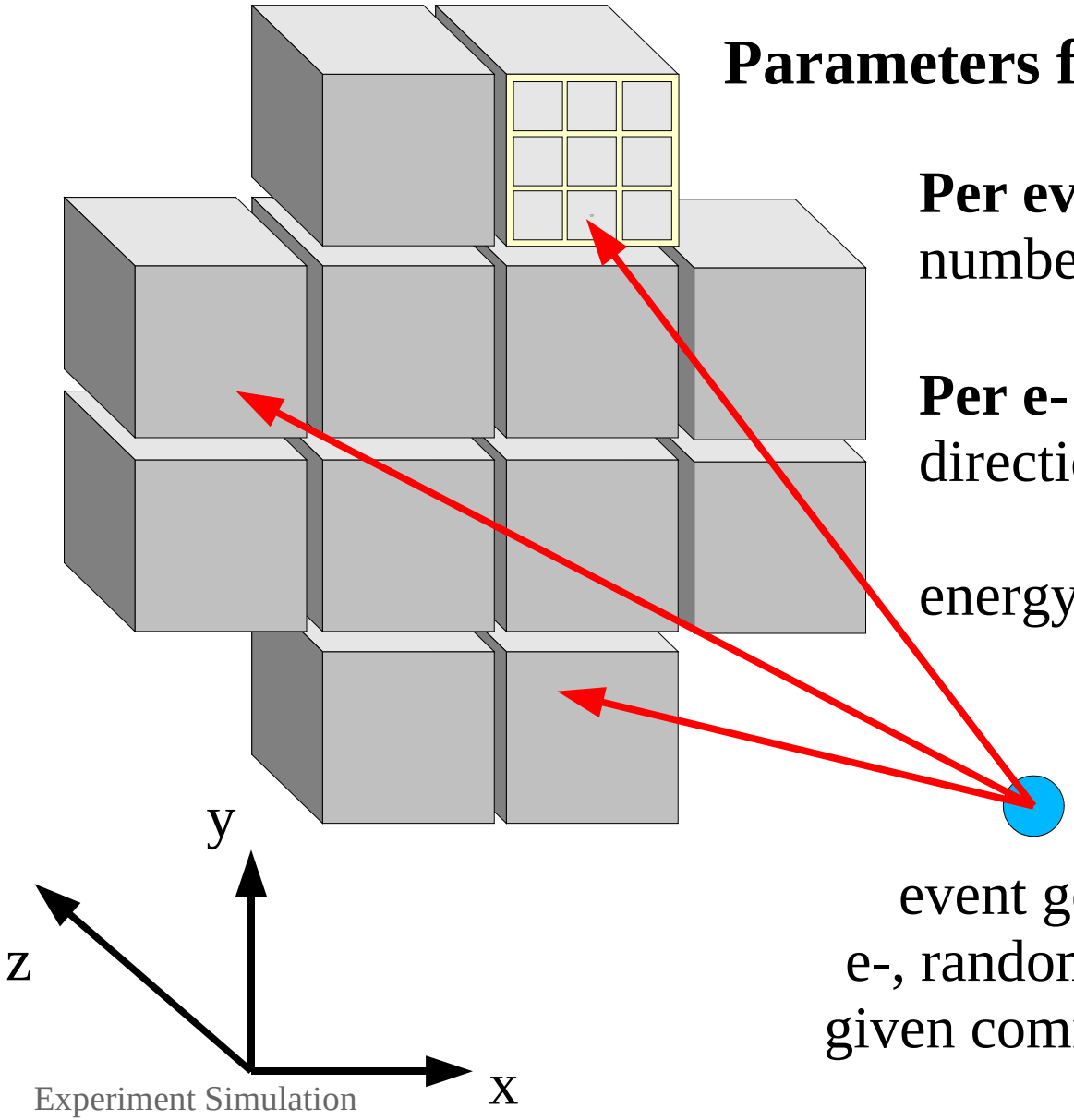


One Endcap has
12 superxtals

Two Endcaps;one Endcap consists of 12 superxtals of given & fixed dimensions in x and y.
z is determined by the length of the contained xtals.

Each superxtal consists of "rows" times "rows" xtals.
The length of a xtal is configurable, "rows" is configurable, and the distance between two xtals ("gap") is configurable.

Primaries



Parameters for the Primary Generator:

Per event at random:
number of e-: [nbPMin,nbPMax)

Per e- at random:
direction: [thetaMin,thetaMax)
[phiMin,phiMax)
energy: n from [nMin,nMax)
e=eStart+eDelta*n

event generator
e-, randomly from a
given common vertex

Simulation Parameters, Metadata: Event Selection Criteria!

We need Simulation Parameters, Metadata to
* to easily study slightly different geometrical/physics setups
* to distinguish samples with same geometrical/physics setups but produced by different teams (random number seeding!!)
* to select persistent hits in our analysis, e.g. all samples where a superxtal containded 3 xtals in a row, ...

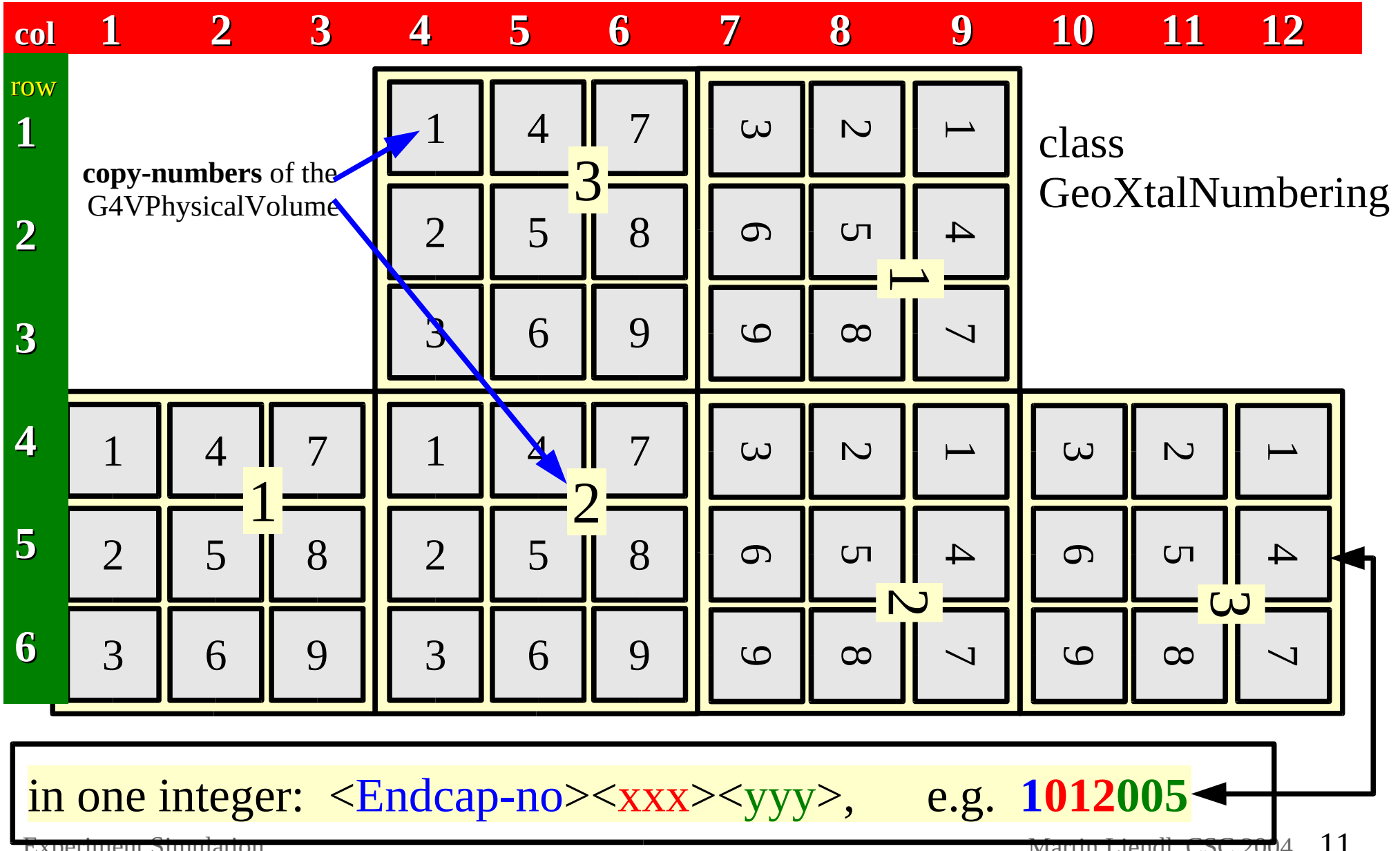
In our exercises, these parameters will be immutable during the execution of one simulation.

We will use:
long **random-seed**; int **rows-of-xtals-in-superxtal**;
string **xtal-material**; double **gap-btw.-xtals**, **half-length-of-xtal**;

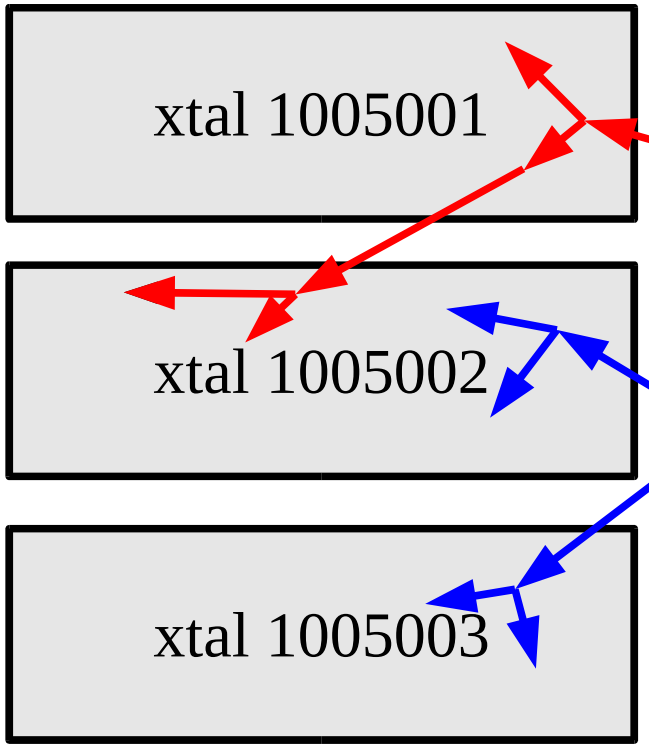
For simplicity, we will **NOT** use:
parameters related to Geant4 physics-lists

Global numbering/addressing scheme for xtals

We need a “global” ID to uniquely identify an xtal!



Hit/Event Definition, Monte Carlo Truth



GeoHit ~ vector<double>
GeoEvent ~ map<HitID,Hit>

Event

HitID		Hit
xtalID	primaryID	E-total, ...
1005001	1	30.2 GeV, ...
1005002	1	15.1 GeV, ...
1005002	2	17.2 GeV, ...
1005003	2	54.8 GeV, ...
...		

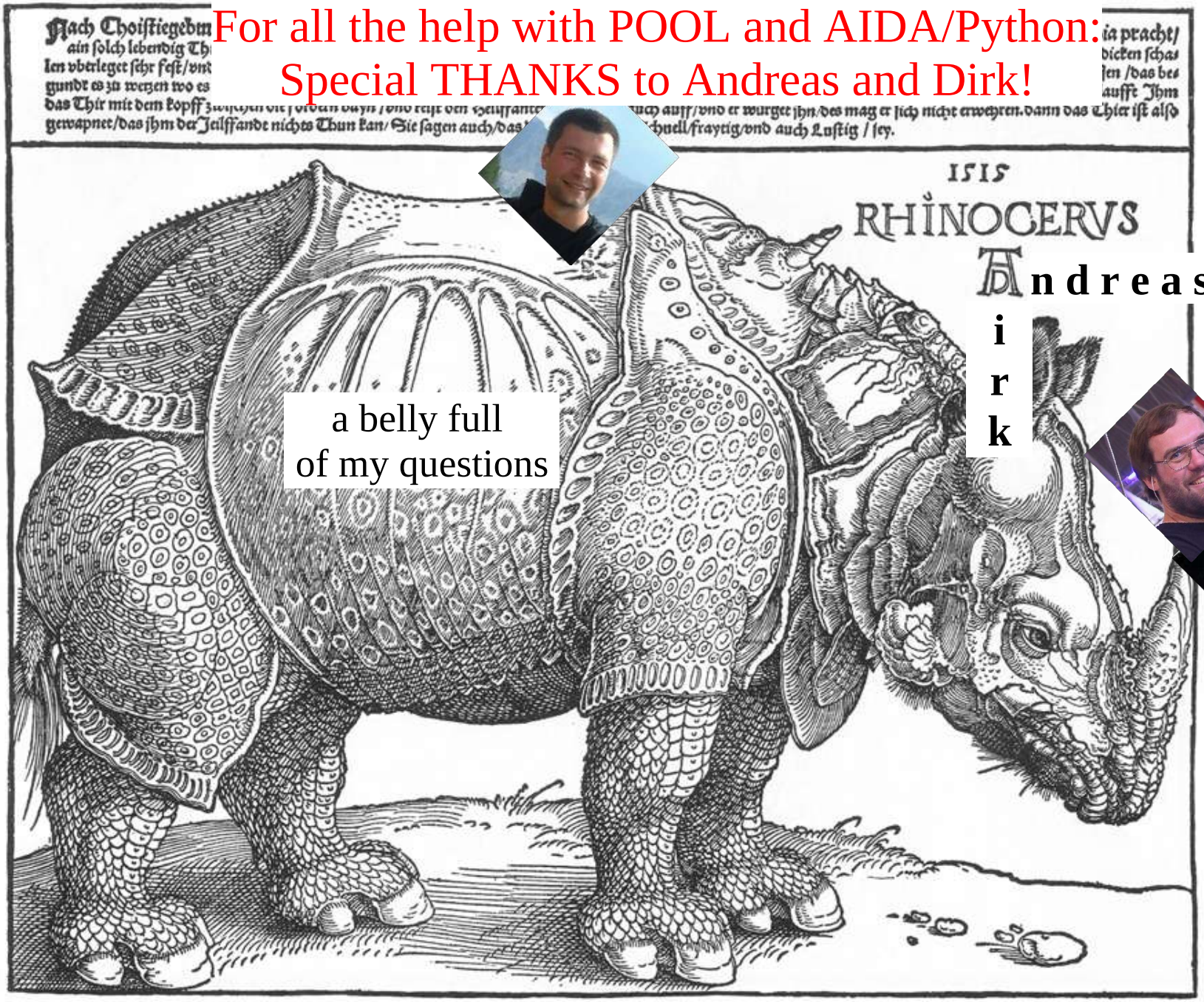
Monte Carlo Truth

primaryID	energy,	direction, ...
1	55 GeV	(0,1,1)
2	90 GeV	(0,-1,1)
...		

```
class GeoEvent;
struct GeoEvent::MCTruth;
class GeoHit;

typedef
map<pair<int,int>,GeoHit>
GeoHitCollection
```

For all the help with POOL and AIDA/Python:
Special THANKS to Andreas and Dirk!



Andreas

Dirk

Getting Started

Login and load this file:

/home/liendl/exercises/start.html

into you browser and follow the instructions!

Select the exercise of your choice

Each exercise consists of the same amount of source files.
Depending on the chosen exercise, some of these files are
not complete => your job to fix this!!!

To get the sources for an exercise, type
(in directory $HOME/simulation/exercises):

./exercise <ex-name> (DON'T FORGET TO PUT THE "./" !!)

where <ex-name> must be one of the following:

no	ex-name	short description	est. difficulty/effort (1...easy, 3...not so easy)
1	esim	run the simulation (it's the solution!)	1
2	primary	implement the primary generator action	1-2
3	detector	complete the detector description	2
4	hits	complete the sensitive-detector hit handling	2
5	numbering	implement a xtal-numbering scheme	2-3
6	ana	implement the analysis	2-3
7	extend	extend the simulation (e+, gamma, pile-up, digis, ..)	3
	SOLUTIONS		
8	analysis	solution of "ana"	
9	aida-ana	same as "analysis" but with results in AIDA-histograms	
10	pool-esim	same as esim but stores data with POOL	

Structure of an exercise <ex>

Root-directory of exercise <ex>:	<EX>:=$HOME/simulation/exercises/<ex>
main() routine defined in:	<EX>/<ex>.cc
Name of the executable:	<ex>
Instructions	<EX>/README

Headers and compilation units:
headers: <EX>/<ex>/include/*.hh
cc-files: <EX>/<ex>/src/*.cc

Simulation macros (more description is contained in the files!):

<EX>/init.mac	Simulation parameters, immutable during an execution of <ex>
<EX>/run.mac	Simulation parameters, mutable between runs in the execution of <ex>
<EX>/vis.mac	Visualization macros for “online” event display called from <EX>/run.mac

File-Conte[n|s]t

Remark: for no obvious reason, most of the source-files are prefixed “Geo”
Contest: Suggest what “Geo” should stand for ...
The team with the best suggestion (best=defined by lectures!)
will be awarded the appropriate amount of beer (or wine, or ...)
(appropriate=defined by me!)
Example: Geo = Generic Electromagnetic Obstacle

Usually, the headers contain commentary of classes and methods
Occasionally, the sources might surprisingly be commented, as well

---

List of relevant sources (omitting .cc, .hh):

<ex>.cc contains the “main()”
instantiates the G4-UserInitializations and -Actions
executes the init.mac, starts the command line

GeoDet defines the geometry

GeoPrimaryGenerator defines the primary generator action

GeoXtalNumbering the numbering scheme for crystals

Files (contd.)

GeoSim keeps track of the simulation parameters from init.mac

GeoHit definition of a hit and a collection of hits

GeoEvent definition of a simulated event and MC Truth

GeoEvents contain GeoHits. GeoEvents are stored persistently!

GeoHitManagement definition of handy Singletons
(the current GeoSim, GeoEvent are made available everywhere in the sources by including GeoHitManagement.hh)

GeoEventIO simple persistency by storing to ascii-files,
event-iterator to loop over stored events for analysis

PoolEventIO will substitute GeoEventIO in or mass-production exercise starting at the end of today!

Generally, you can safely ignore the other files. Commentary in the sources which you should modify will point you to other code when relevant ...

Geant4 documentation

/home/liendl/docu/g4/html/index.html

Compile & execute

To compile & link: in sim_exercises/<EX>/<ex> simply type **make**

To execute: simply type <ex>
(csh: a **rehash** might be necessary)

Testing

Once you get your exercise running, it even might produce some output!

A simple looper over the stored events is available by building the exercise “check”

```
cd ...../sim_exercises
./example check
cd check
make
check <seed> <rows> <xtal-material> <gap> <half-length>
```

(parameters of the simulation runs – as provided in init.mac)

check will loop over data stored in $HOME/sim_data

check will use the utility **GeoEventPrinter** to print an ASCII representation of the xtal-matrix and the energies contained in the xtals of each GeoEvent.

Instructions for the exercises

The following slides contain documentation for the exercies. Choose one exercise, read some documentation and try to solve it!

GOOD LOOK!

Together, we will do exercise 1 (“esim”) right now!

Exercise 1 “esim”:
Run the simulation with varying parameters

go to directory $HOME/simulation/exercises/esim
type **make** and wait until the library and executable are built.

open the README file & also read it
(each exercise will have it's own REAME with instructions!!!)

open the file “init.mac” & read the commentaries
open the file “run.mac” & read the commentaries
open the file “vis.mac” & read the commentaries

-> discussion of the parameters; change some parameters in init.mac
or run.mac

start a simulation, type: esim
some events should be simulated according to your init.mac/run.mac

Exercise 1

after the first couple of simulated event, a command prompt should appear

on the command prompt:
change some parameters of the primary generator
(use commands as seen in run.mac)

run again some events by typing: /run/beamOn <nb-of-events>

Check the results:

cd .../sim_exercises
./example check
cd check
make
check <seed> <rows> <xtal-material> <gap(mm)> <half-length(mm)>
(use the parameters as specified in init.mac!)

Exercise 2 “primary”: Implement the primary generator action

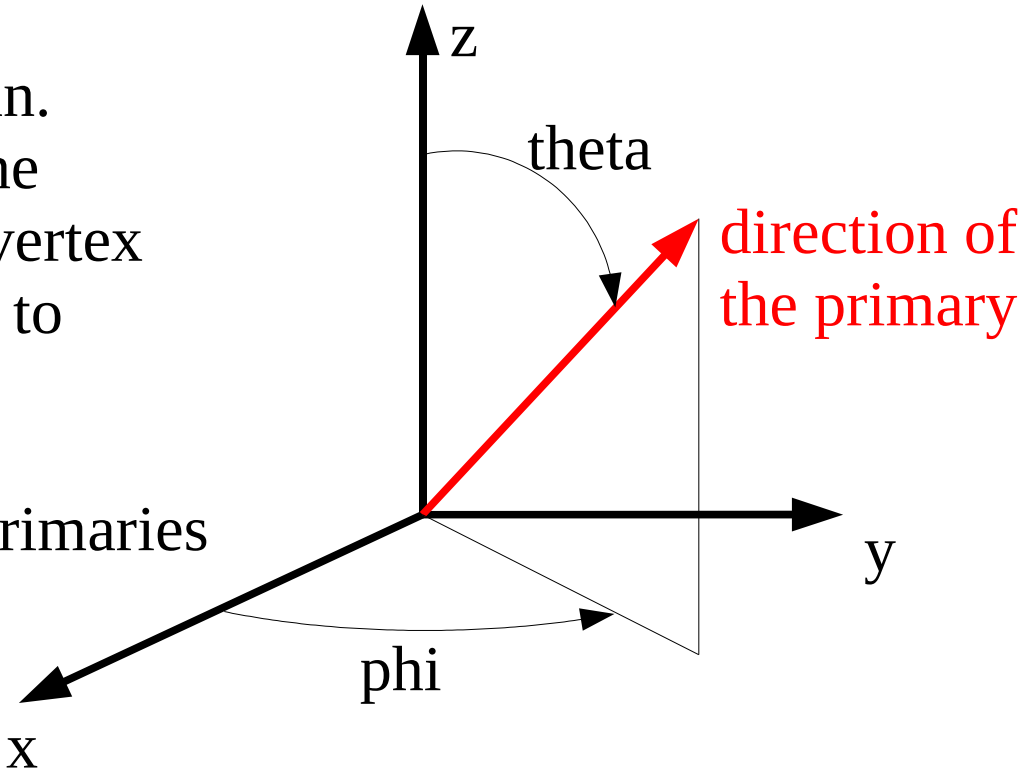
Primary particles must be added to a G4Event
G4 calls the GeoPrimaryGenerator-action
at the beginning of each event.

Geant4 provides you a G4ParticleGun.
The particle gun can be used to set the particle type, direction, energy, and vertex position of one primary and assign it to a G4Event.

Use the G4ParticleGun to generate primaries at random the ranges as specified on the next page!

code: GeoPrimaryGenerator.hh,cc

definition of angles:



Exercise 2

Call to G4UniformRandom() produces a random numbers uniformly distributed in [0,1).

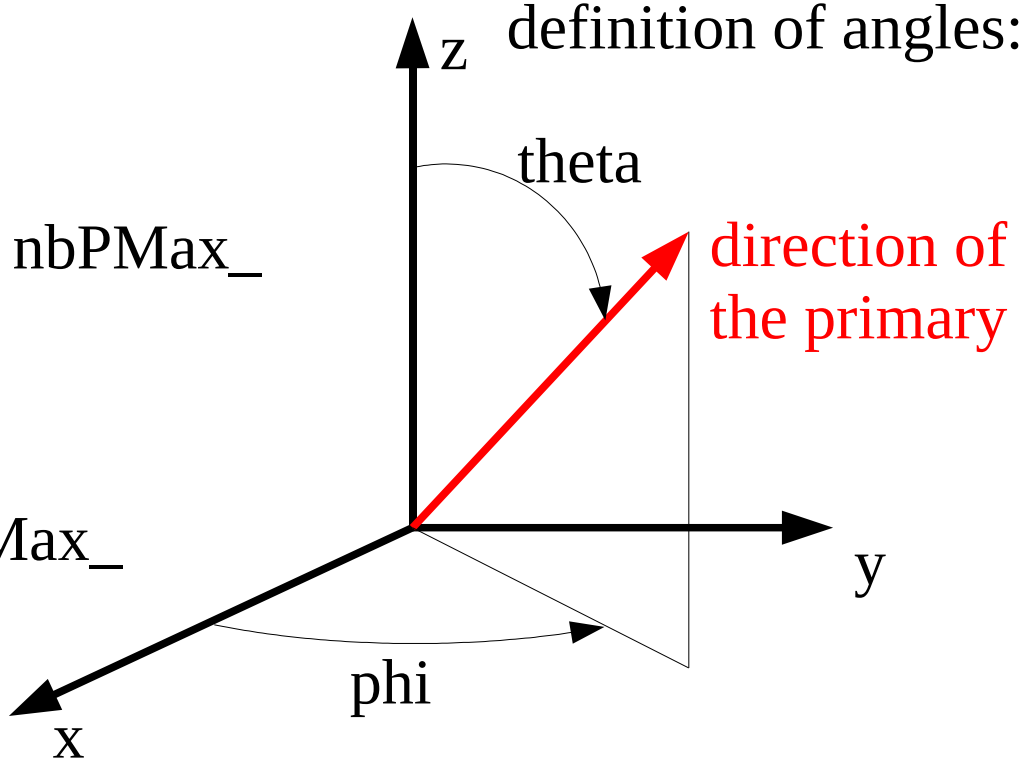
The ranges for primary generation are defined in GeoPrimaryGenerator.hh

Per event at random:
number of e- between nbPMin_ and nbPMax_

Per e- at random:
<u>direction:</u>
theta between thetaMin_ and thetaMax_
phi between phiMin_ and phiMax_

<u>energy:</u>
n between nMin_ and nMax_
energy = eStart_+eDelta_* n (eStart_, eDelta_ are fixed)

definition of angles:



Exercise 3 “detector”: Complete the Geometry

In this exercise you should complete the geometry
of the electromagnetic calorimeter.

Your task is to fill the superxtals with xtals by respecting the following parameters:

rows_ ... rows of xtals in a superxtal
gap_ ... gap between xtals
xtalHalfLength_ ... half length of an xtal
material ... the material, the xtals is made of

Crystals are modeled as boxes.

code: GeoDet.cc

The following pages contain the necessary details of the geometry ...

Exercise 3

Geometry setup for CSC simulation exercises:

- Two **Endcap** electromagnetic crystal calorimeter make up the detector.
- One Endcap disc is made of 4 **HalfDee** quarters of a disc.
- Each HalfDee contains three boxes, the **superxtals**.
- Each superxtal contains a quadratic grid of **xtals**.

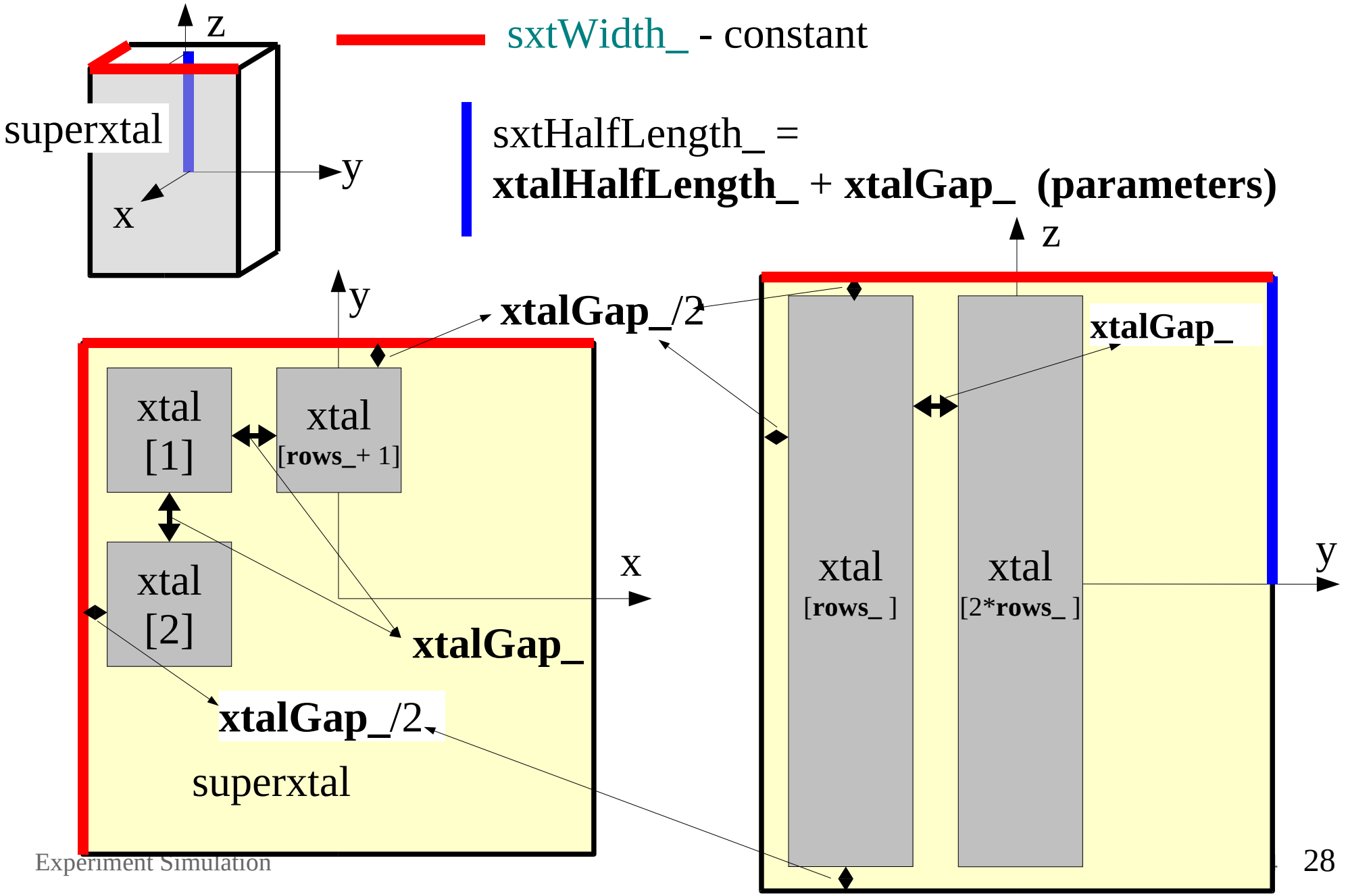
The Endcap calorimeter is parameterized by three parameters:

xtalHalfLength_ ... half length of one xtal (see next pg.)

rows_ ... number of xtals per row in the superxtal

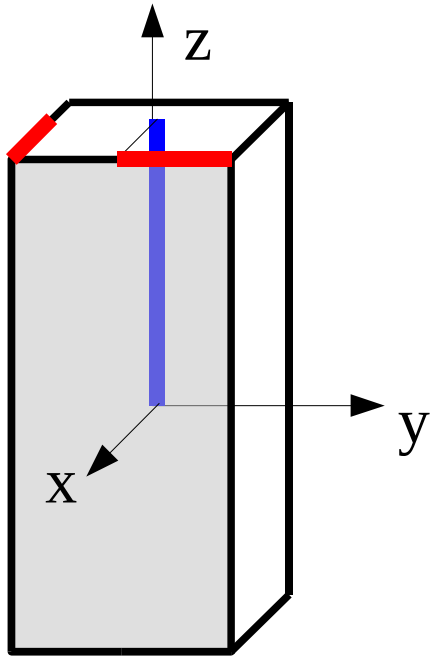
xtalGap_ ... space between two xtals

superxtals in the electromagnetic calorimeter (sxt ... superXtal) **cise 3**



Exercise 3

Crystals in the electromagnetic calorimeter

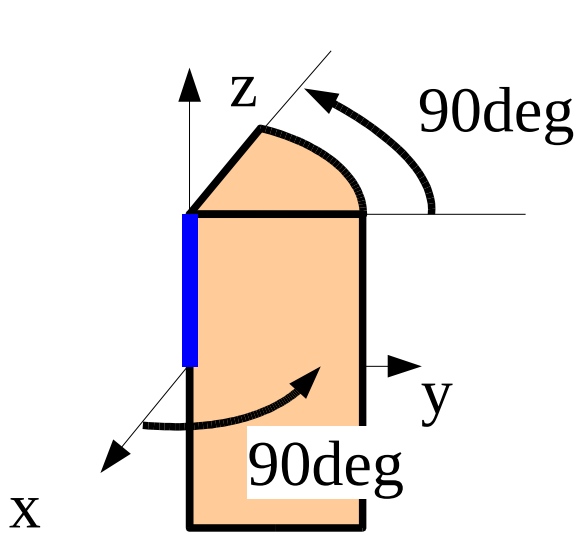


xtalHalfWidth_ = (sxtWidth_/**rows_ - xtalGap_**)/2

xtalHalfLength_

"xtal", G4Box

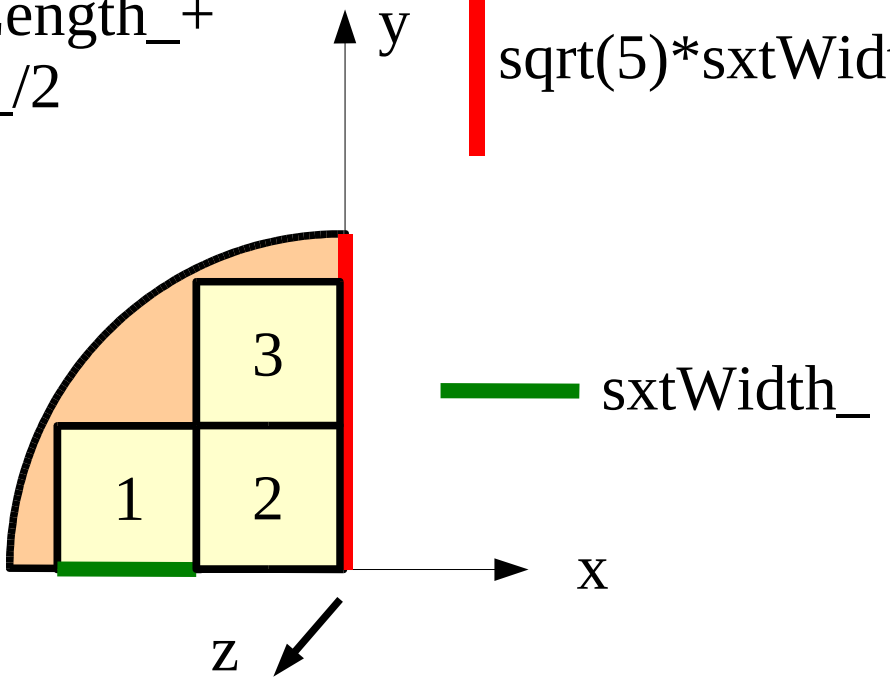
“HalfDee” of “superXtals” in the electromagnetic calorimeter — Exercise 3



HalfDee

sxtHalfLength_+ xtalGap_/2

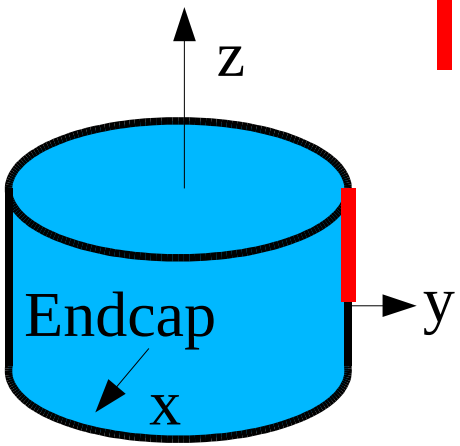
sqrt(5)*sxtWidth_



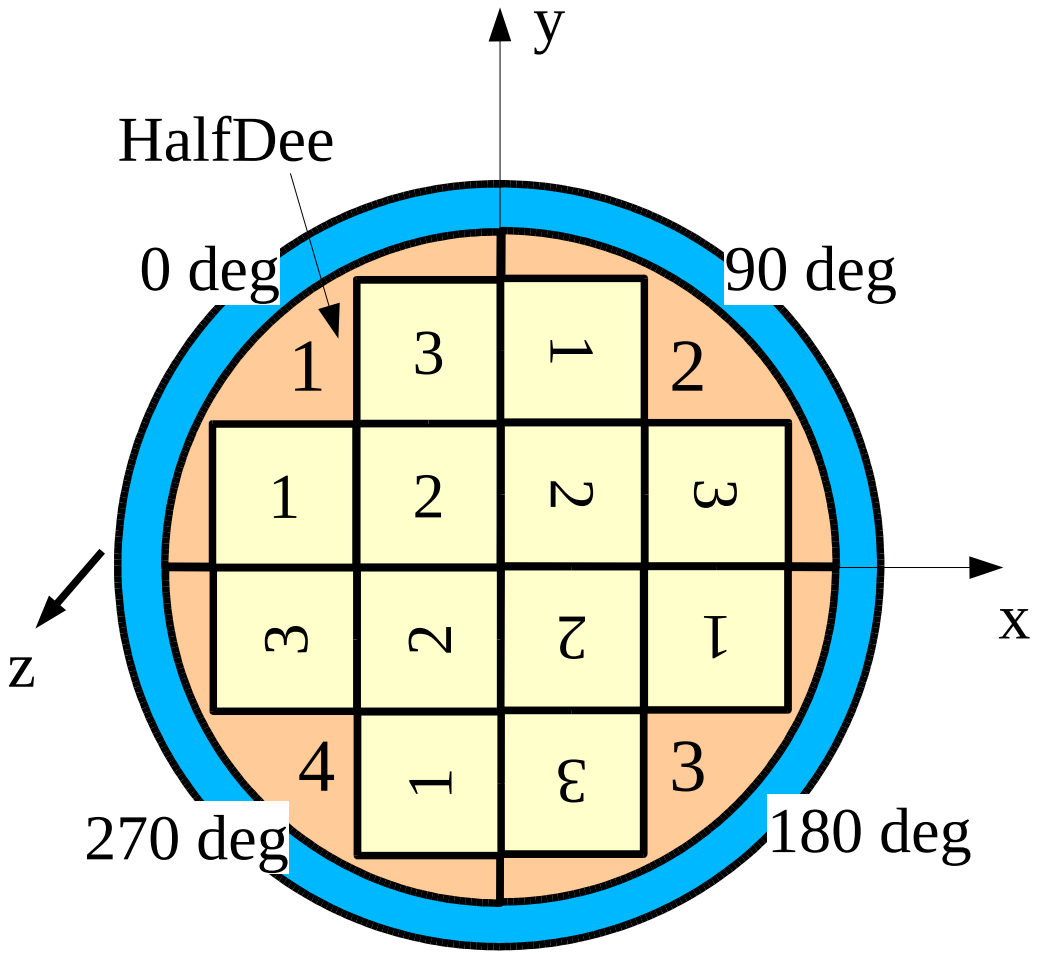
sxtWidth_

Exercise 3

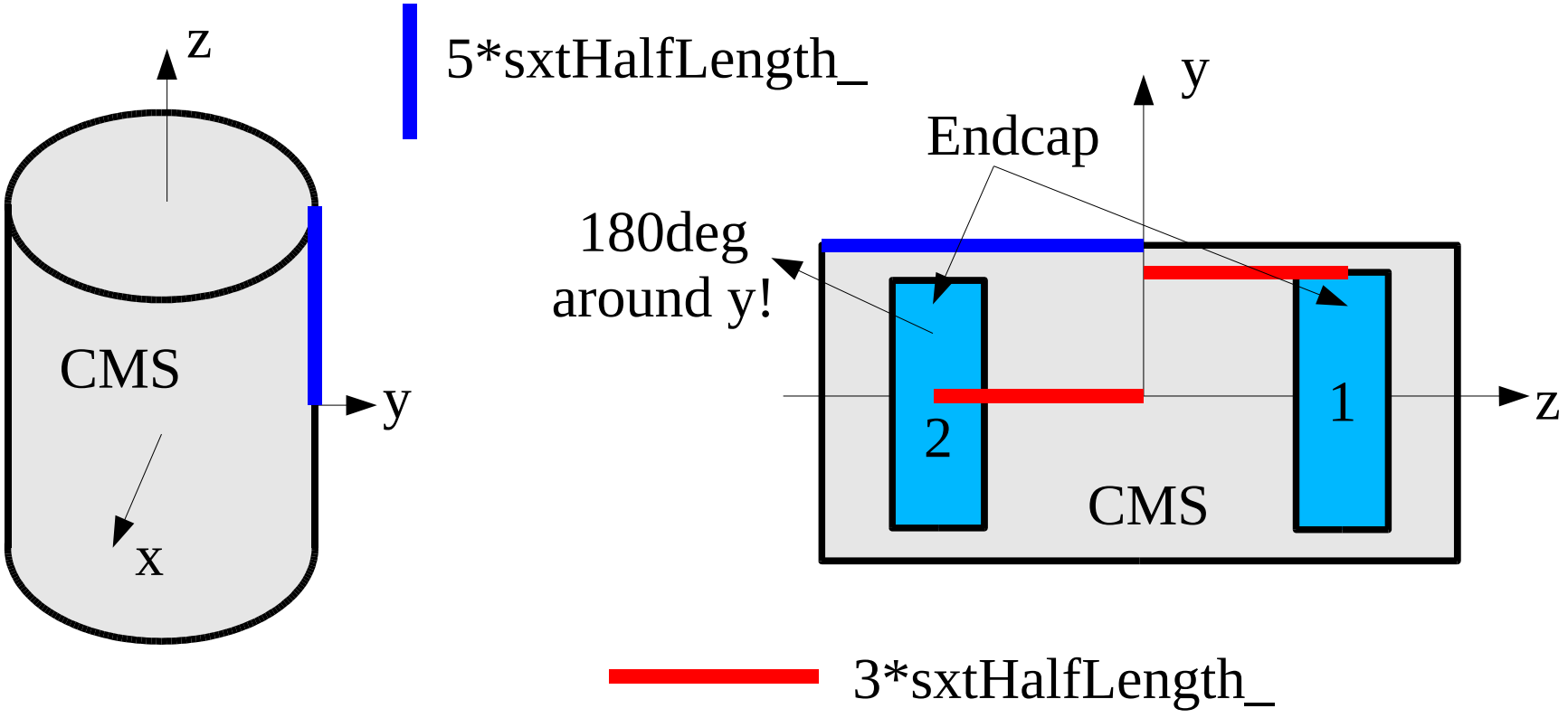
Endcap



sxtHalfLength_+
xtalGap_/2



CMS (with only 2 Endcaps of the electromagnetic calorimeter) – Exercise 3



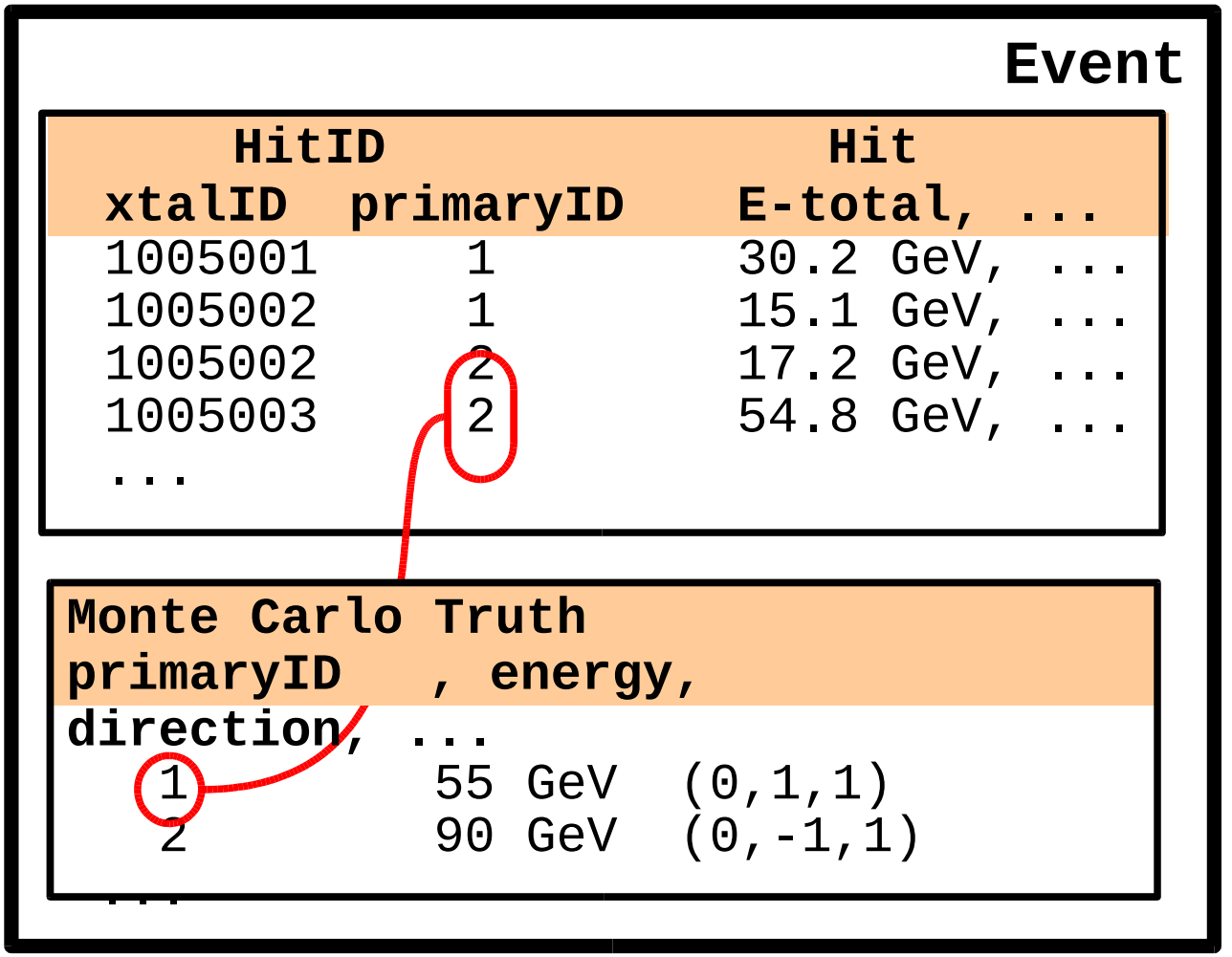
Exercise 4 “hits”: Implement the Sensitive Detector Action

Remember:
each step in a sensitive volume will trigger the ProcessHits-method of the sensitive-detector-user-action!

Study the hit and event definitions in GeoHit.hh, GeoEvent.hh

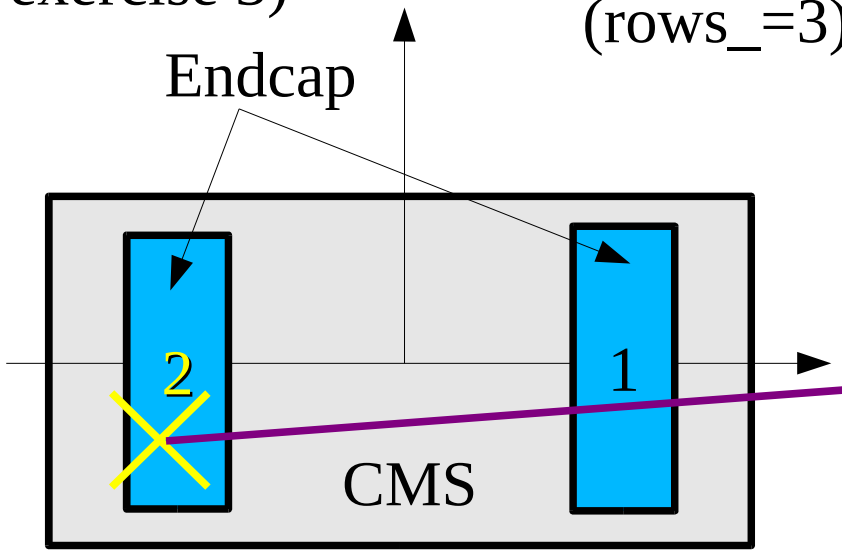
Complete GeoSensitive.cc

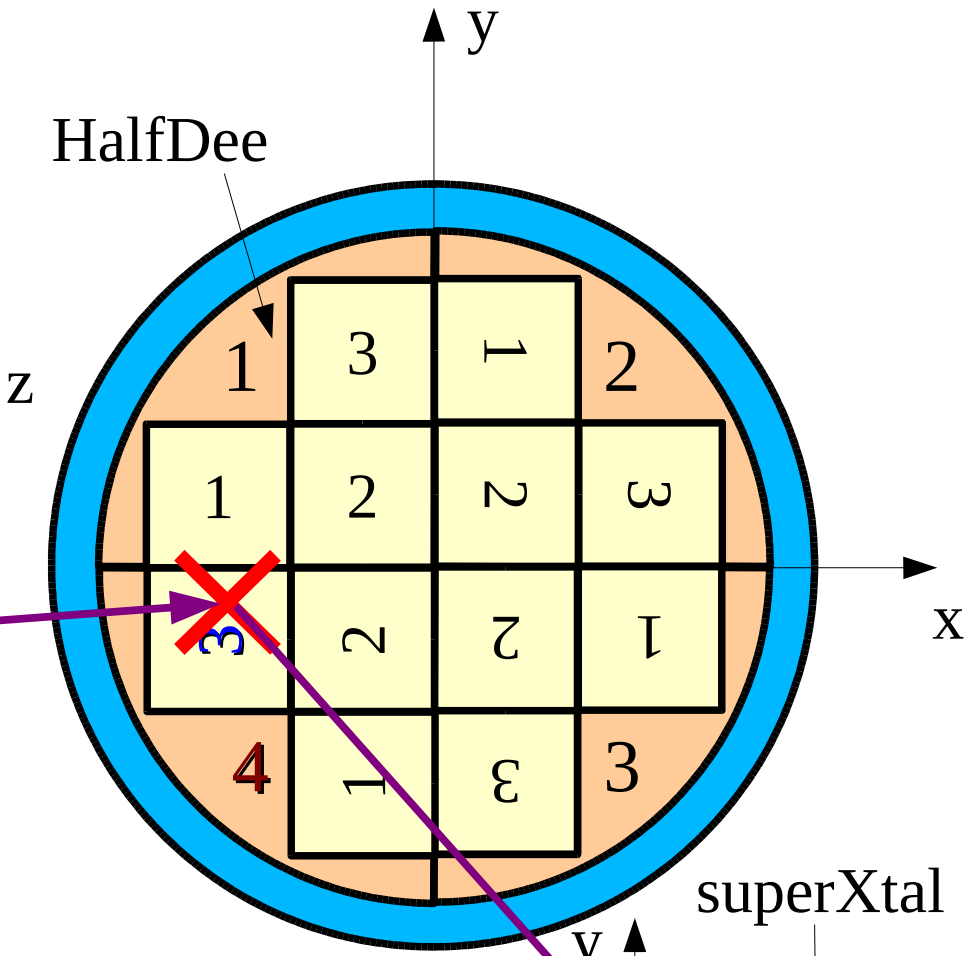
Make the xtals sensitive in GeoDet.cc



Exercise 5 “numbering”

(for details on the geometry structure:
see exercise 3)

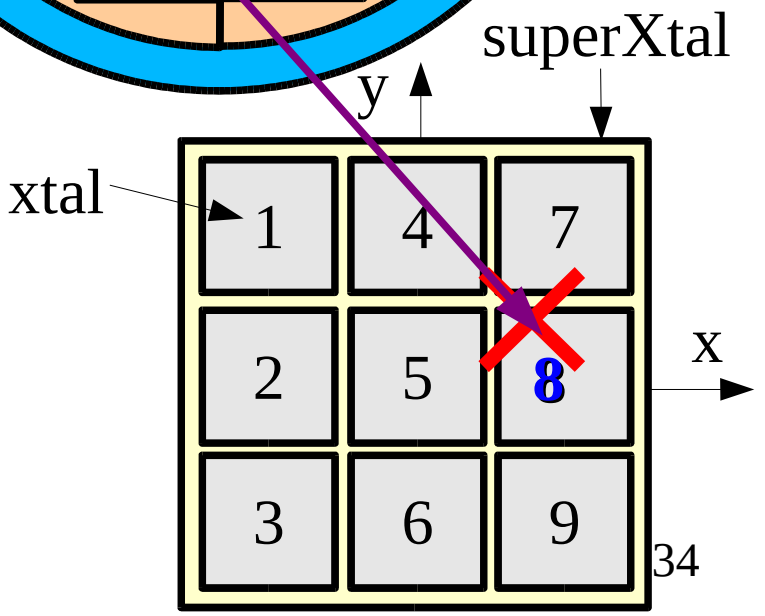




during hit-processing you get
from the G4VTouchable and the
copy-numbers of the G4VPhysicalVolumes:
stack of physical-volumes

CMS	1
Endcap	2
HalfDee	4
superXtal	3
xtal	8

xtalID = f(2,4,3,8)



Exercise 5

Simple numbering scheme:

```
int xtalID(int ecap, int halfdee, int superxtal, int xtal) {
    // format: ehsxx  (e..endcap-no, h...halfdee-no, s..superxtal-no, xx..xtal-no)
    return ecap*10000 + halfdee*1000 + 100*superxtal + xtal;
}
```

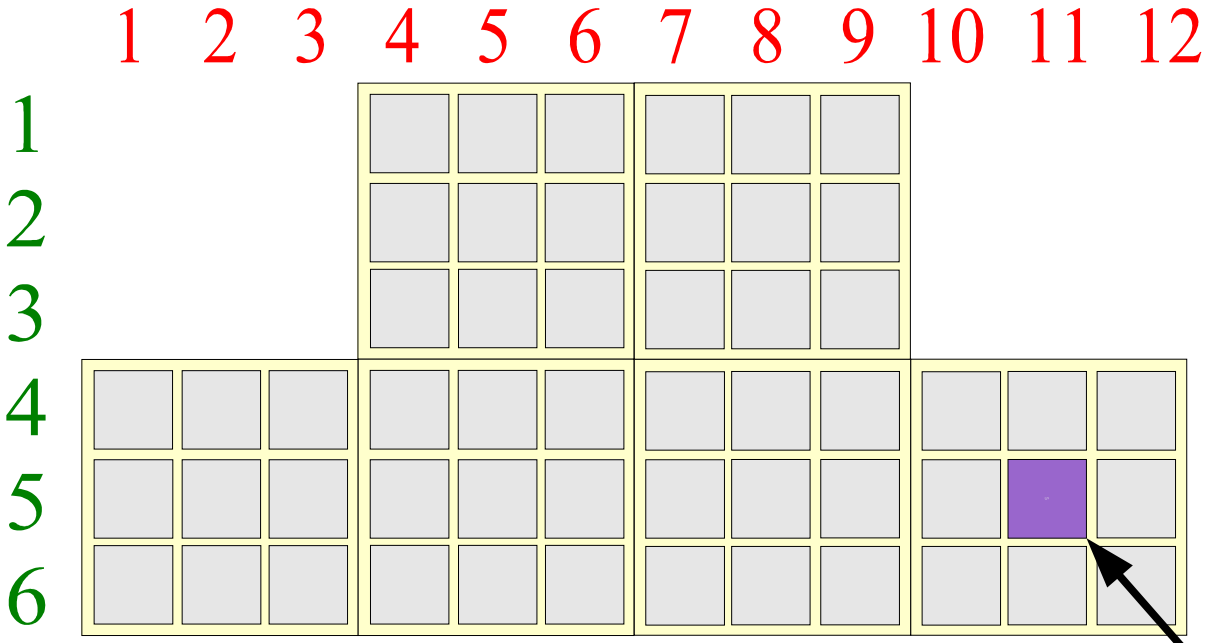
It's hard to “navigate” this numbering scheme, i.e. are
xtal 11103 and xtal 11104 neighbors?

The numbering scheme does provide an x-z “view” on the crystals.
It merely reflects the topological graph structure of the geometry.

You can implement this numbering scheme, but it will be incompatible with our analysis routines!!!

“Navigable” numbering scheme

Exercise 5



You have to implement this scheme to be compatible with the analysis code!

code:
class GeoXtalNumbering

column row

integer: <Endcap-no><xxx><yyy>, e.g. **1011005**

Exercise 6 “ana”: Implement the analysis

If you are already familiar with Geant4:

use “esim” from exercise 1 to produce a bigger sample of events while you are implementing the analysis.

Study GeoHit.hh, and GeoEvent.hh for the hit- and event-information.

Implement the analysis-loop in UserAnalysis.cc

Instructions for the analysis task are in the last chapter “Analysis”.
The solution for this analysis is available as “./example analysis”

Modify the Hit structure and save more information per hit in “esim”. You can use this additional information in a more extensive analysis. **Example: find out about the average duration of a shower depending on the incident energy ....**

Exercise 7 “extend”: Extend the simulation

If you are already familiar with Geant4:

Implement digitization with some little pile-up (per event):
- extend GeoHit with pile-up information, i.e.
  . add a field that stores the overall energy deposited in the xtal
  . add a field for the ADC-count corresponding to the digital signal proportional to the deposited energy
- assign an ADC count with some noise to the total energy deposited in one xtal
- think of MC Truth information (nb. of primaries, their energy) as information from the tracker
- simulate a sample
- modify “./example analysis”: try to find well-separated clusters in the ADC-patters ...

Exercise 8 “extend”: Extend the simulation

This is the fully solved analysis !

Exercise 9
Weekend-production "pool-esim":

Each group should build **./example pool-esim**

Then edit init.mac

>> parameters to be provided during the exercise <<

Then edit vis.mac

>> parameters to be provided during the exercise <<

Then start "pool-esim" an let it run until it finishes some time during this night or tomorrow ....
(data in $CSC_DATA/<seed>_<rows>_<material>_<gap>_<hlenght>)
Check tomorrow, if it has not crashed ...

Monday we will combine the event data and run a global analysis!

Analysis: Energy Resolution

For Monday! Please read during the WE!

The shower development is a statistical process. This explains why the relative accuracy of energy measurements in calorimeters improves with increasing energy, according to the formula:

$$\sigma/E \sim a/\sqrt{E}$$ (for simplicity: other terms ignored)

E ... energy of the incident particle
σ ... standard deviation of energy measurement

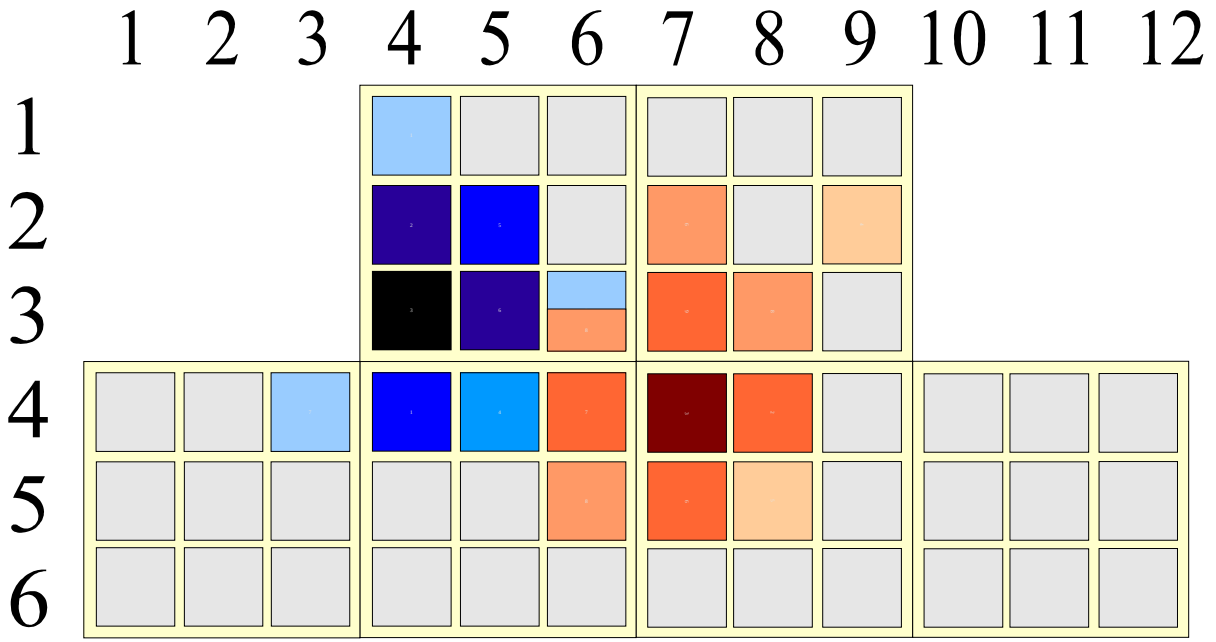
Calculation:
Provided we have n primary particles i (i=1,..n) of the same energy.

E_i ... energy deposited by the i-th particle (from our simulated hits)

$$E_{avg} = 1/n * \Sigma E_i \quad ; \quad var = 1/(n-1) * \Sigma (E_{avg} - E_i)^2$$

$$\sigma = \sqrt{var}$$

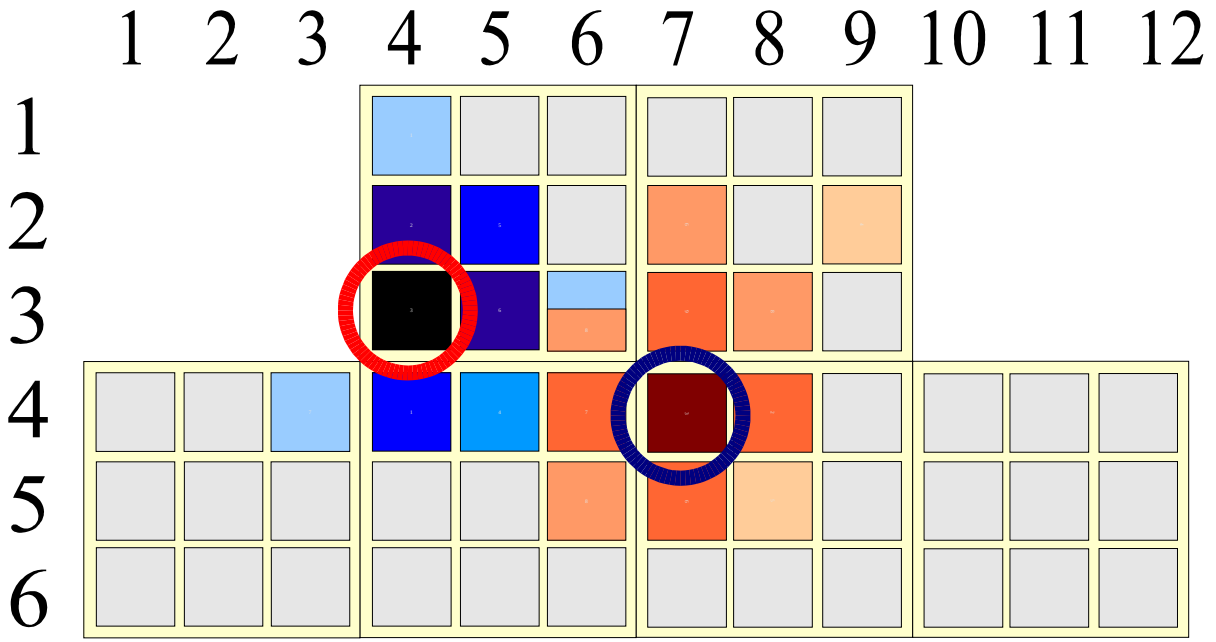
Analysis: Energy Resolution



One Event:
from MC Truth: two particles (e-), with different energies
from Hits: energy deposit of each particle per xtal

Note: in our example we CAN distinguish between the energies of multiple particles in the same xtal. Not obvious in reality!

Analysis: Energy Resolution

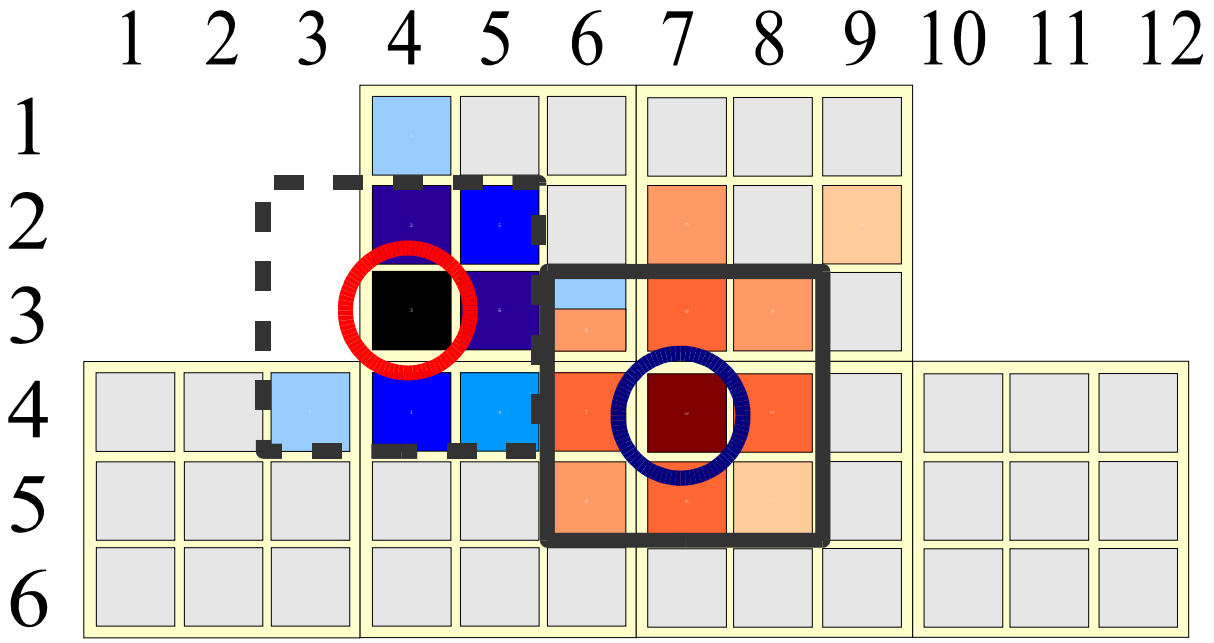


Step one:

find the xtals where each primary lost most of its energy (loop over the hits-collection of one event)

One Event:
from MC Truth: two particles (e-), with different energies
from Hits: energy deposit of each particle per xtal

Analysis: Energy Resolution



Step two:

Sum up the energies in windows of 3x3 xtals around the xtals with max. energies. Ignore, if the 3x3 window is incomplete:

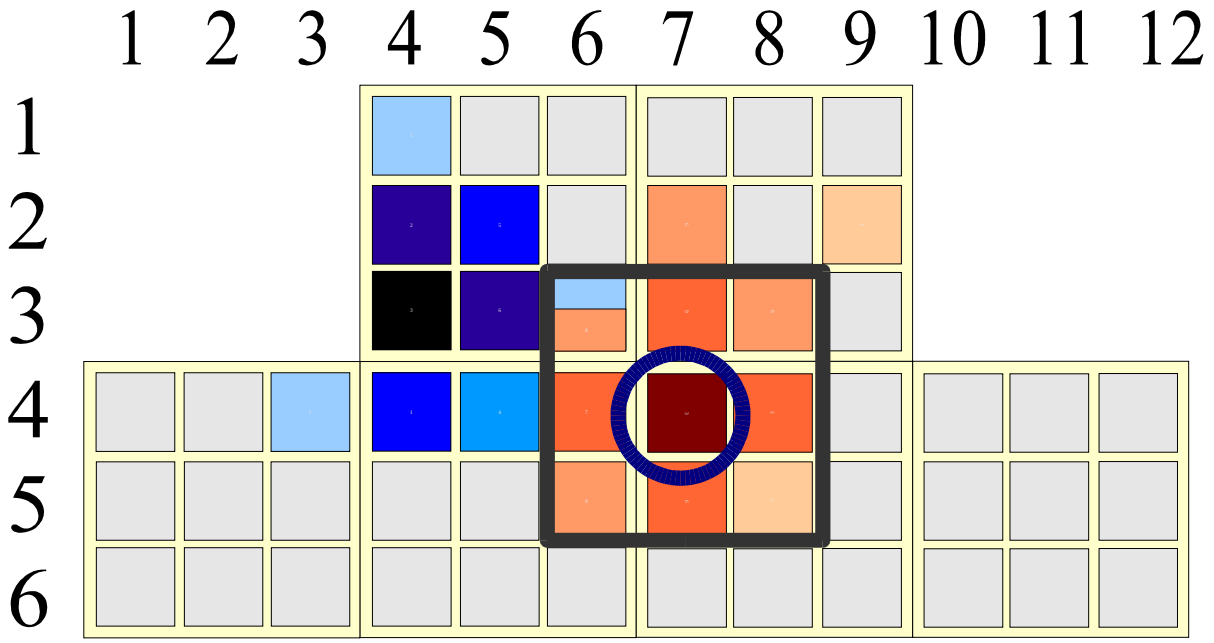
=> defines E_i

for each energy of the primaries.

$$E_{avg} = 1/n * \Sigma E_i \quad ; \quad var = 1/(n-1) * \Sigma (E_{avg} - E_i)^2$$

$$\sigma = \sqrt{var}$$

Analysis: Energy Resolution



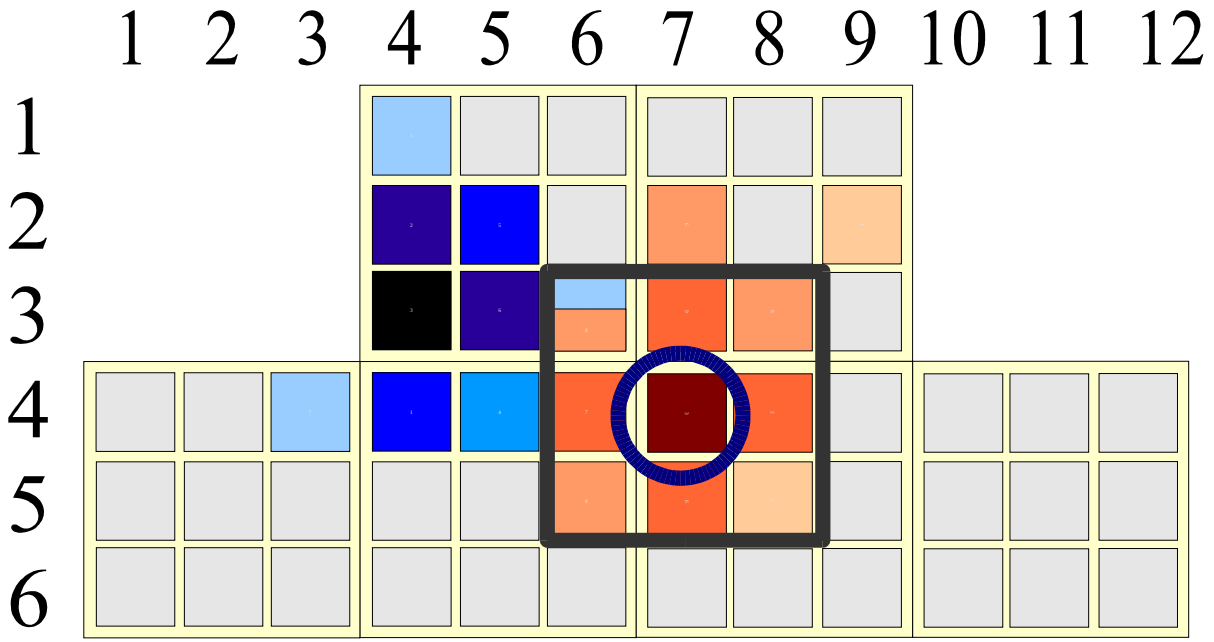
Step three:

Foreach 3x3 cluster E_i:
Look in MC Truth for the true energy $E_T(i)$ of the particle. Put E_i into a histogram H_T => determines E_{avg} and σ **for each** incident energy!

$$E_{avg} = 1/n * \sum E_i \quad ; \quad var = 1/(n-1) * \sum (E_{avg} - E_i)^2$$

$$\sigma = \sqrt{var}$$

Analysis: Energy Resolution



Analysis Loop:

Repeat Steps 1-3
for each event in the persistent store
that has been produced with the same simulation parameters
(except for the random number seed!!)

What for?

Check resolution of different geometrical setup
Check the linearity: true energy vs. 3x3 mean
Compare with testbeam-data
...

Analysis-code

The code for the analysis can be found in

./example analysis (for the ASCII-based persistency)

and the same analysis code for POOL based persistency:

./example pool-analysis

THANK YOU ALL!!!

:-)