The University of Manchester

LArSoft simulation

Anyssa Navrer-Agasson

Slack: #simulation

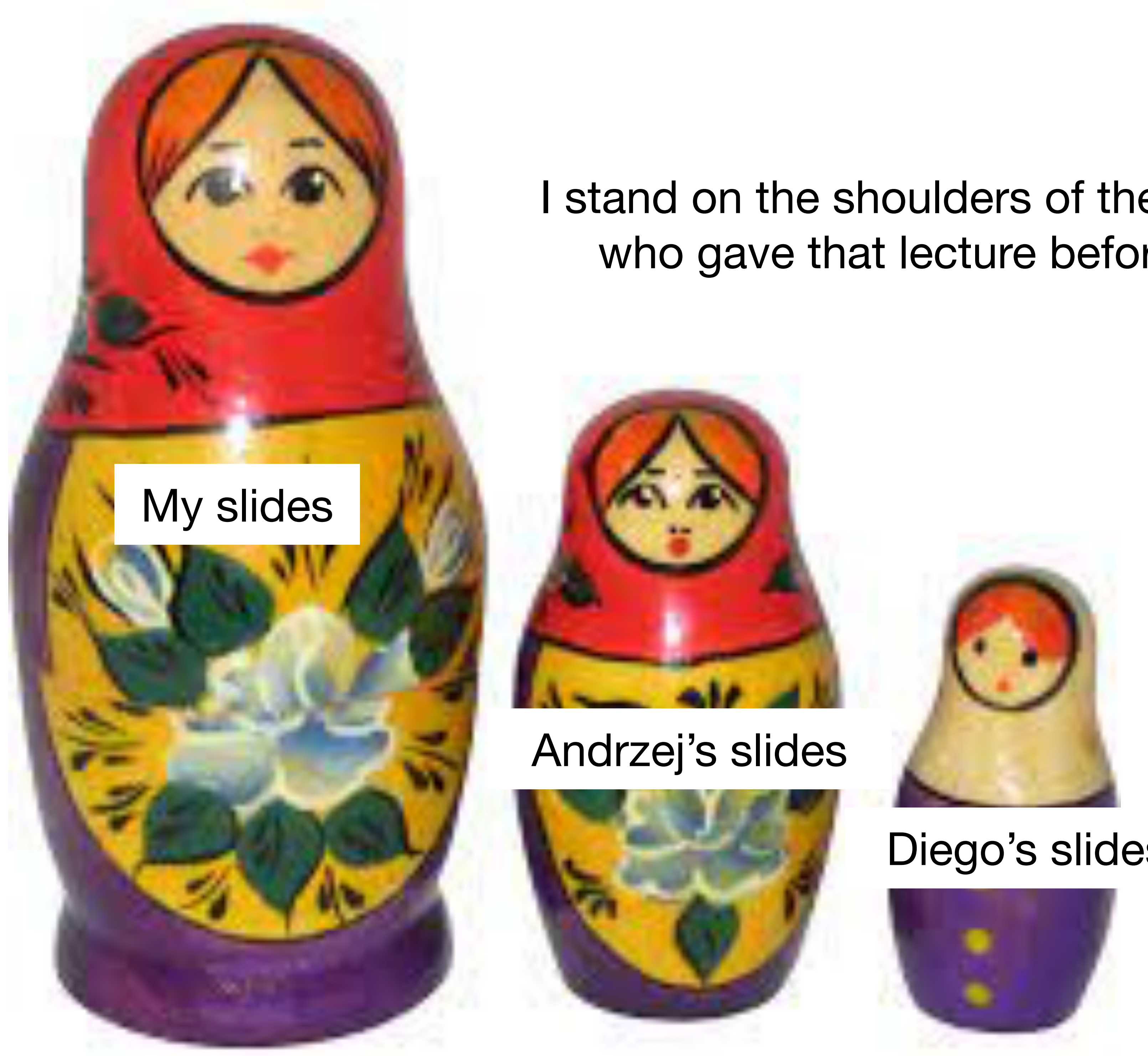
UK/Latin America LArSoft Workshop - 7th September 2022

Goal of the lecture

Aka what will you know in 1h?

- What are the steps needed to generate events?
- What are the different tools used for each step?
- How do different part of the simulation communicate?
- What is the output of each step?

I stand on the shoulders of the people who gave that lecture before me



Simulating events with LArSoft

MANCHESTER 1824
The University of Manchester

- General FNAL LAr experiments simulation framework:
  - Only need to learn one framework, even if you're working on multiple experiments.
  - Need to have both common and experiment specific parts.
- In the following lectures/tutorials you will learn about how to reconstruct events. This lecture will help you understand how these events get generated.
- This helps to understand why the reconstruction needs to do what it needs to do.

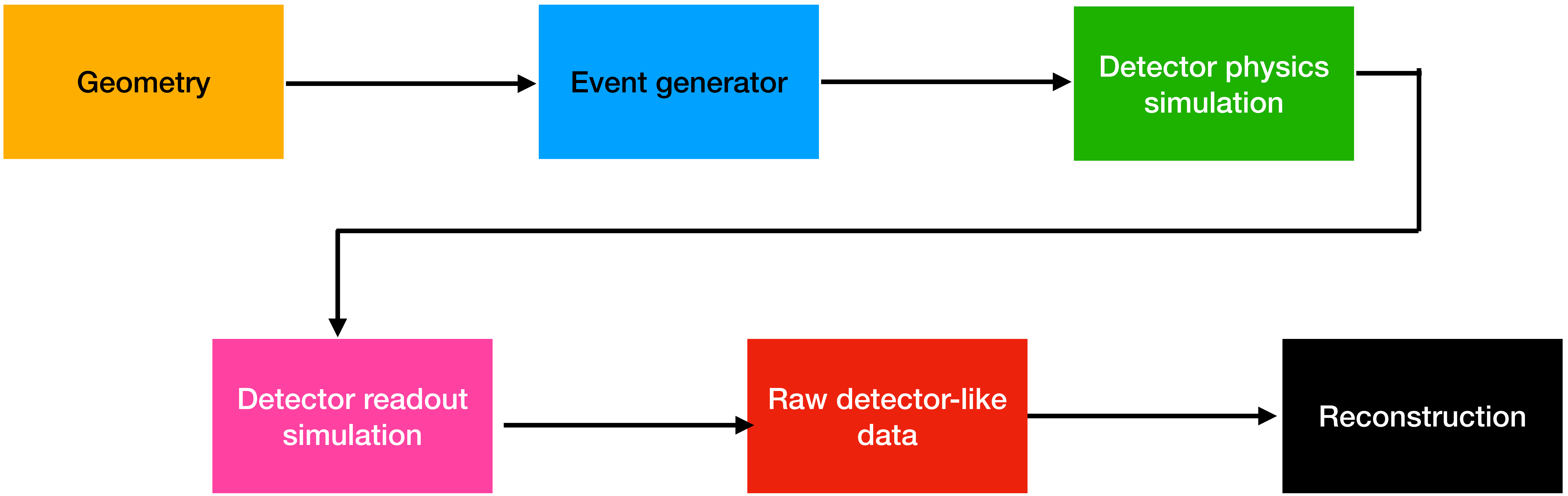
What are we trying to do?

The University of Manchester

- Produce events that look like real data, but with “truth” information to check the behaviour of the reconstruction/analysis.
  - Output should have the same format and contain the same information as real data.
  - Simulation needs to be affected by the detector response.

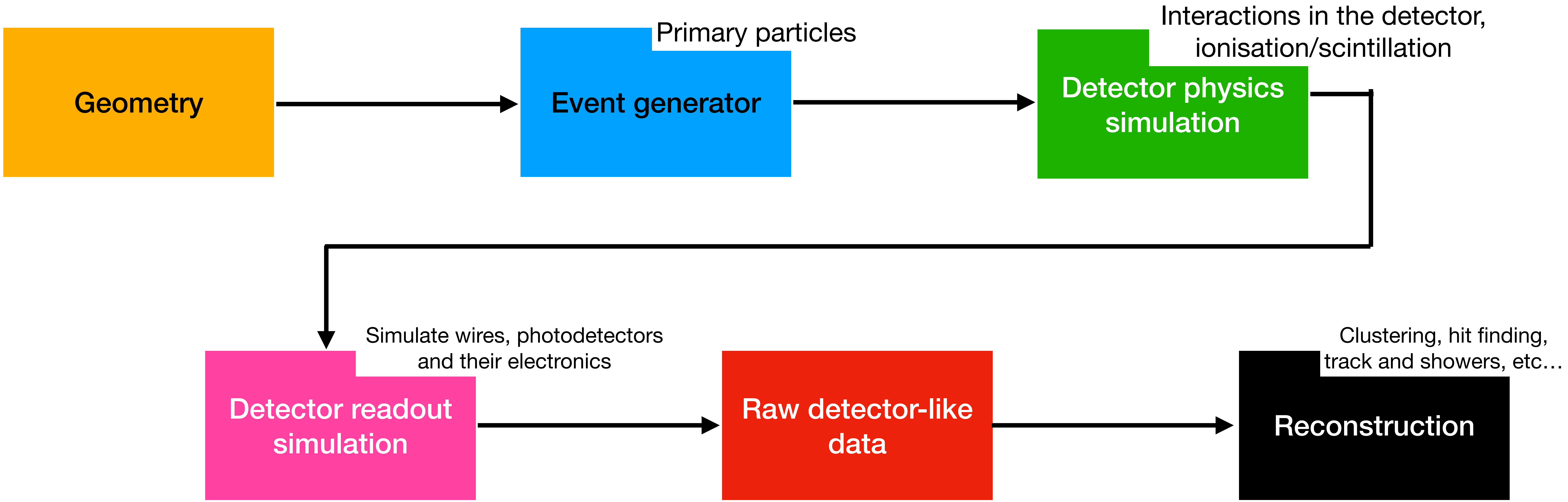
LArSoft simulation flowchart

Steps



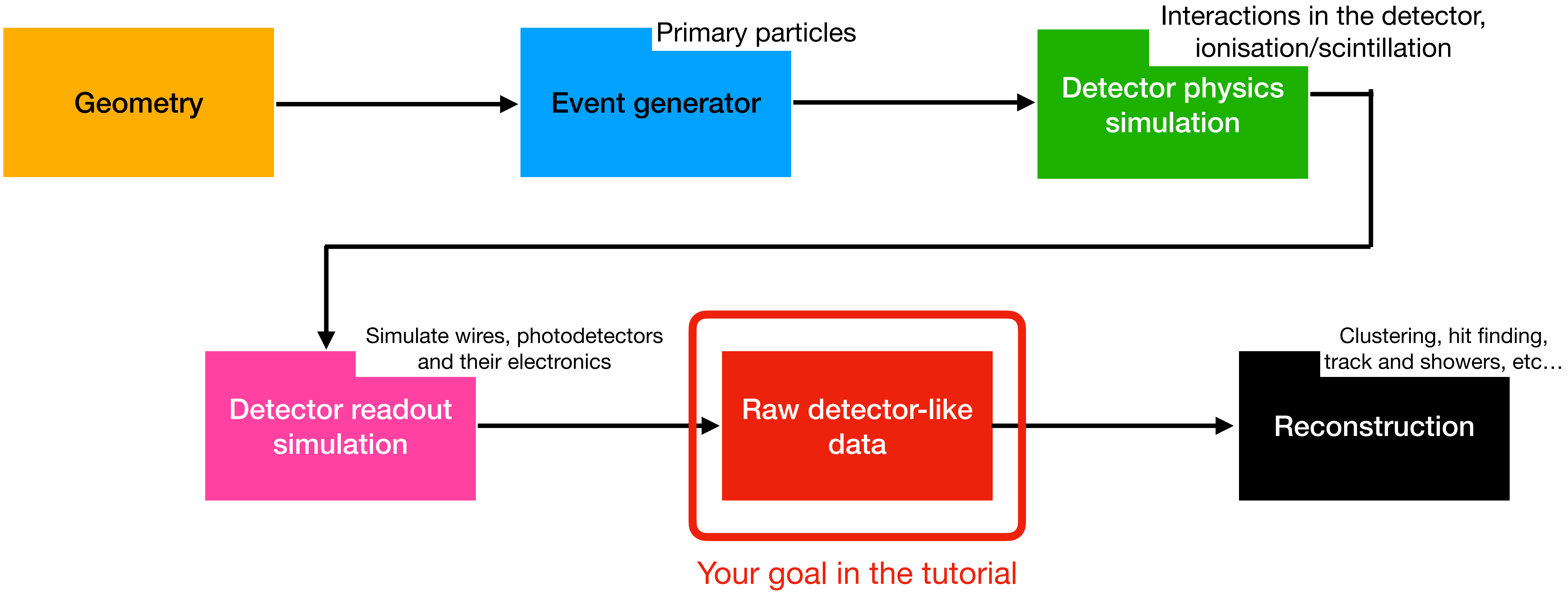
LArSoft simulation flowchart

Steps



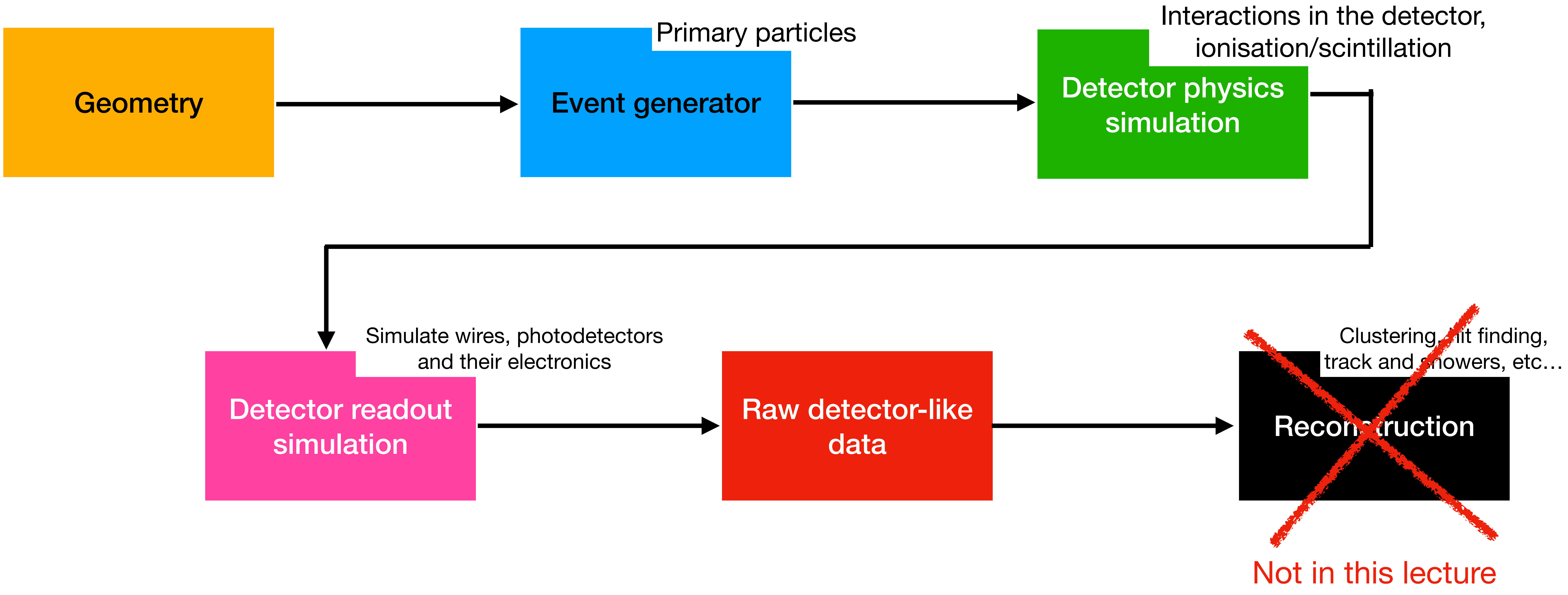
LArSoft simulation flowchart

Steps



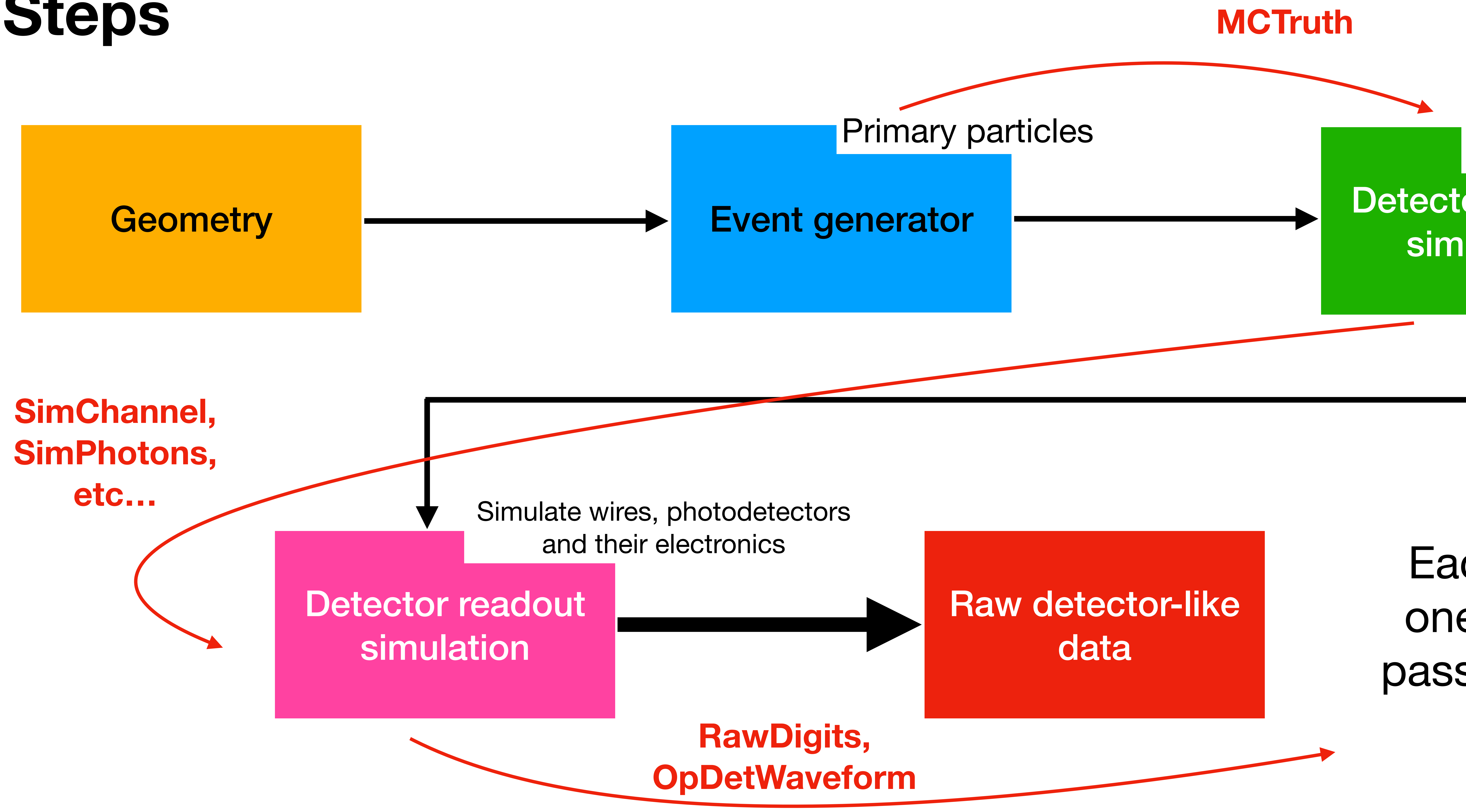
LArSoft simulation flowchart

Steps



LArSoft simulation flowchart

Steps



Each stage is composed of one or more “modules” and passes “data products” to the next.

LArSoft simulation flowchart

MANCHESTER 1824
The University of Manchester

Communication: services

- Services are classes with only one instance managed by the framework and can be accessed by the different modules.

- They provide information about (non-exhaustive lists):

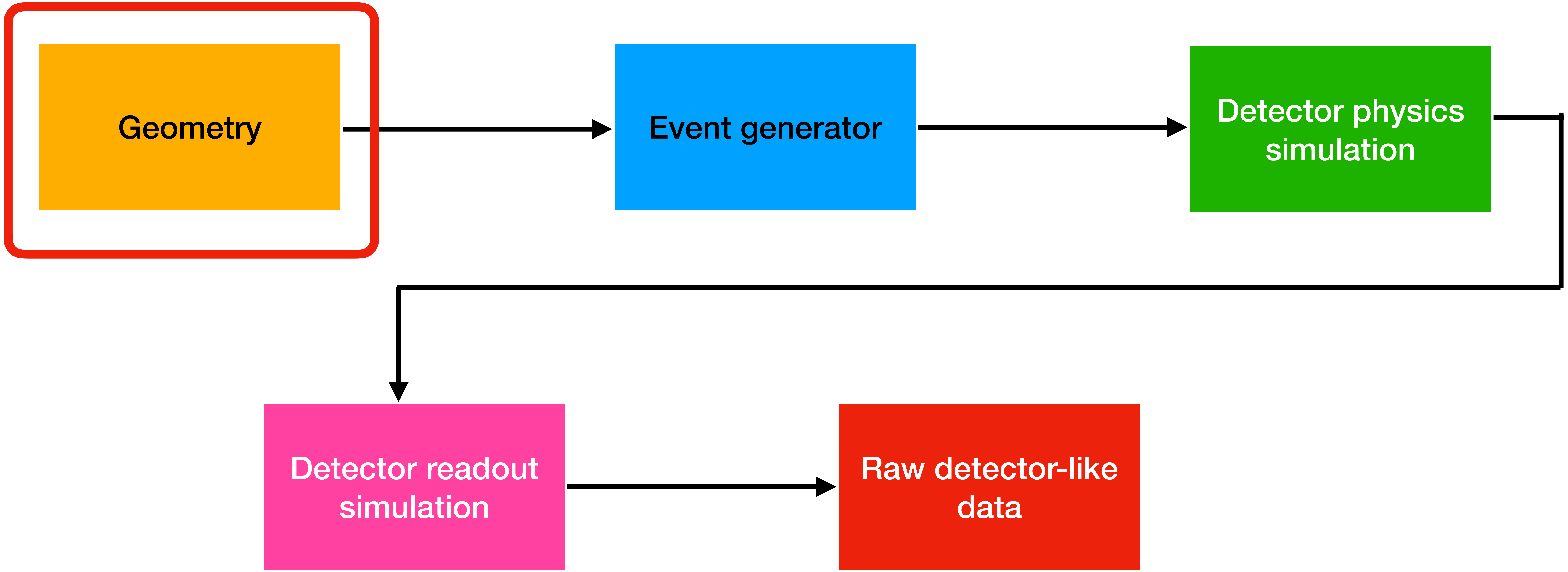
  - Geometry: TPC structure, optical detectors positions, auxiliary detectors (e.g. CRT)

  - Physical properties: LAr properties (i.e. radiation length), detector properties (i.e. drift velocity)

  - Physics simulation: GEANT4 parameters

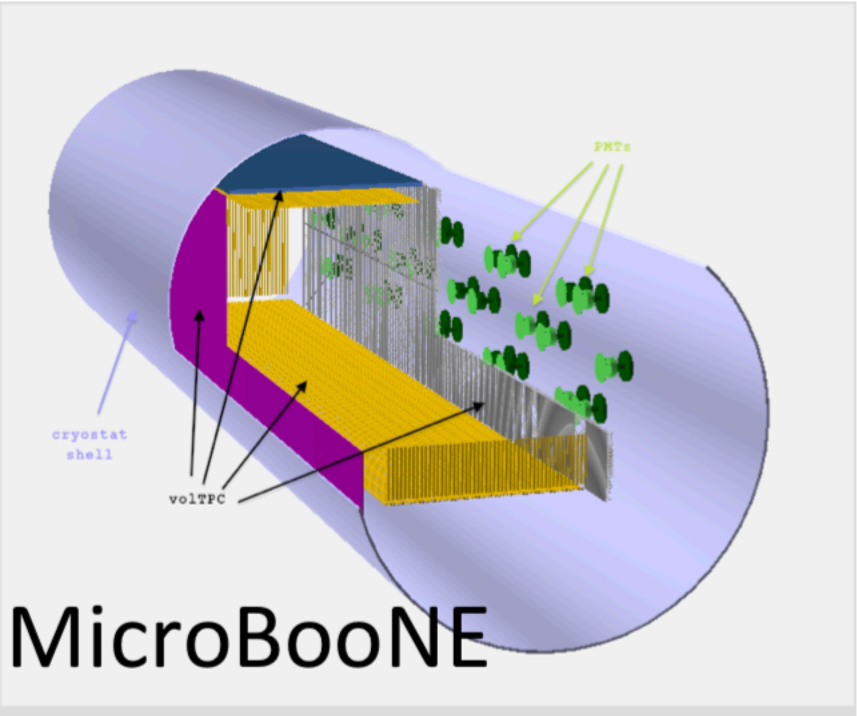
LArSoft simulation flowchart

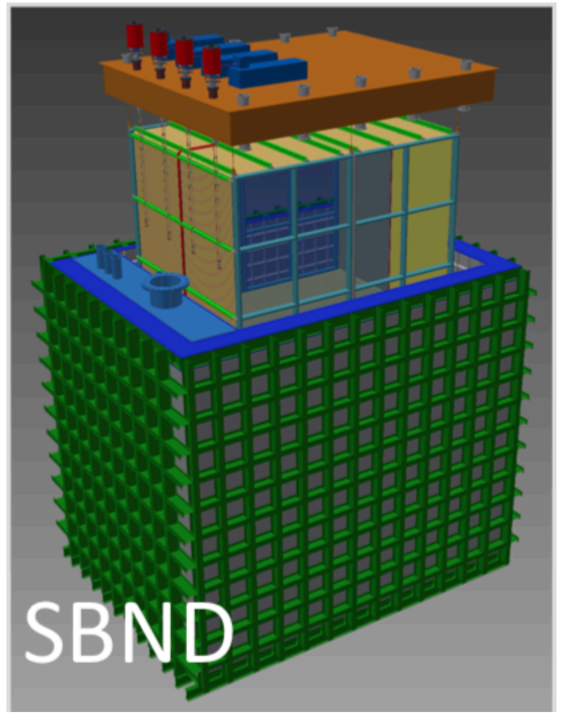
The University of Manchester

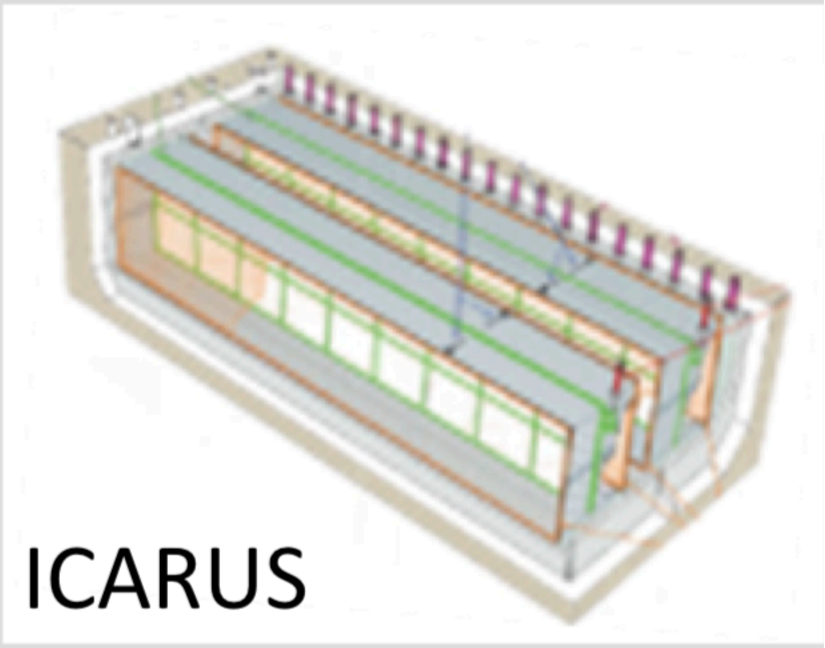


Geometry

MANCHESTER
1824
The University of Manchester





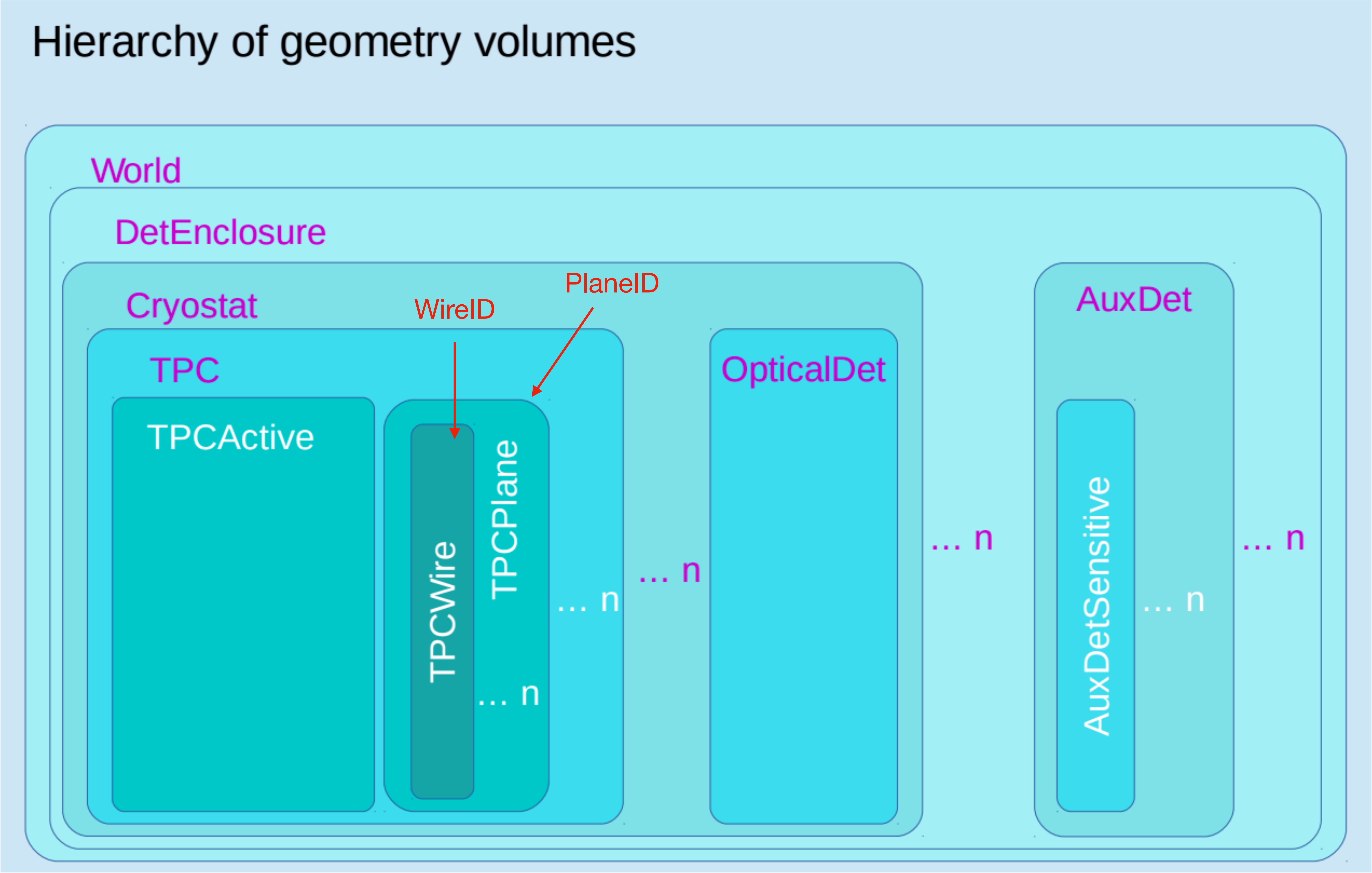


and more!

- Each detector just needs to add a new geometry description
- Simulation/reconstruction knows how to access different geometries, but are not dependent on any one
- Uses GDML (Geometry Description Markup Language)
- Detectors have two versions of the geometry:
  - With wires: used to determine wire location and properties
  - No wires: actually used in simulation (saves time and memory)

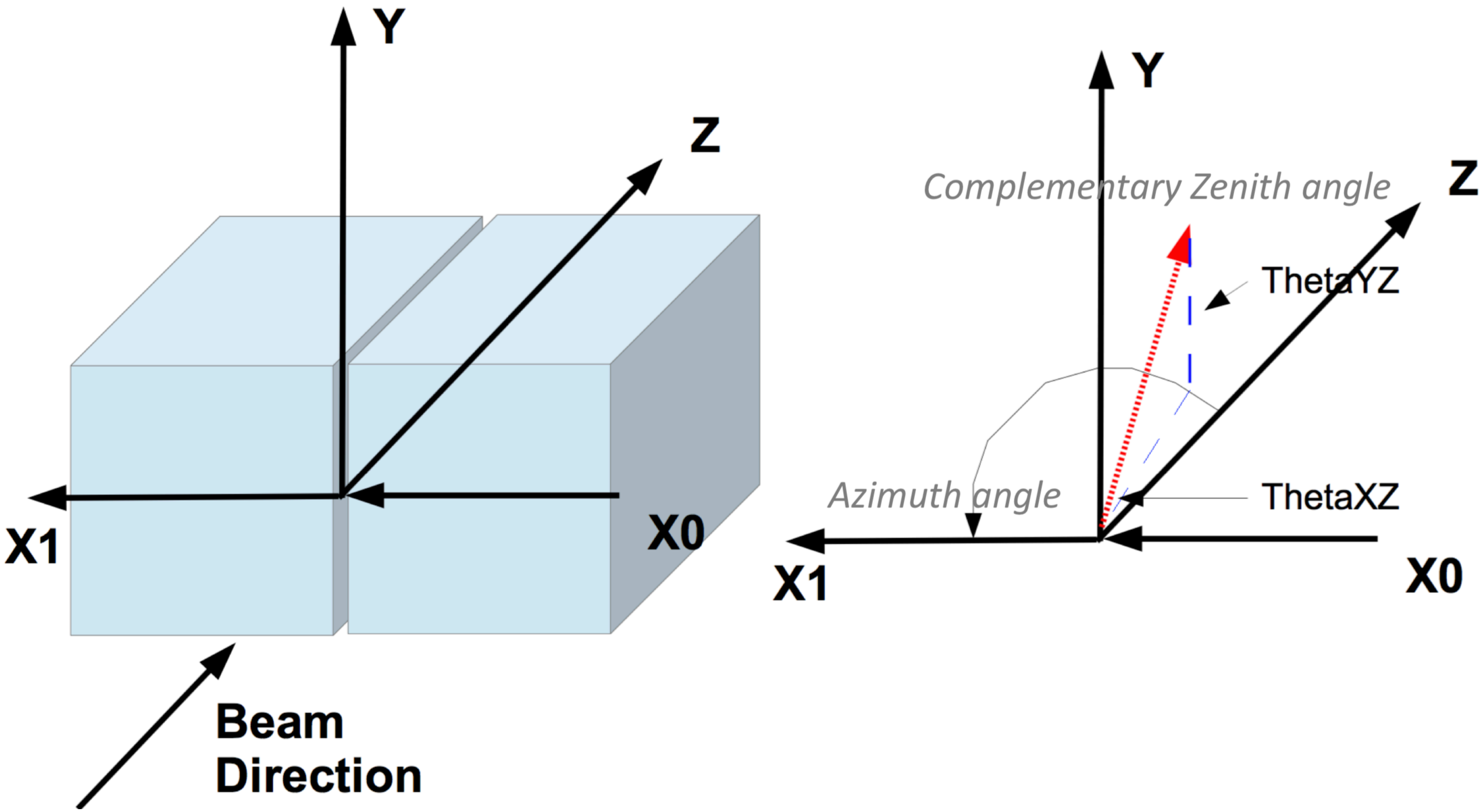
Geometry model in LArSoft

The University of Manchester



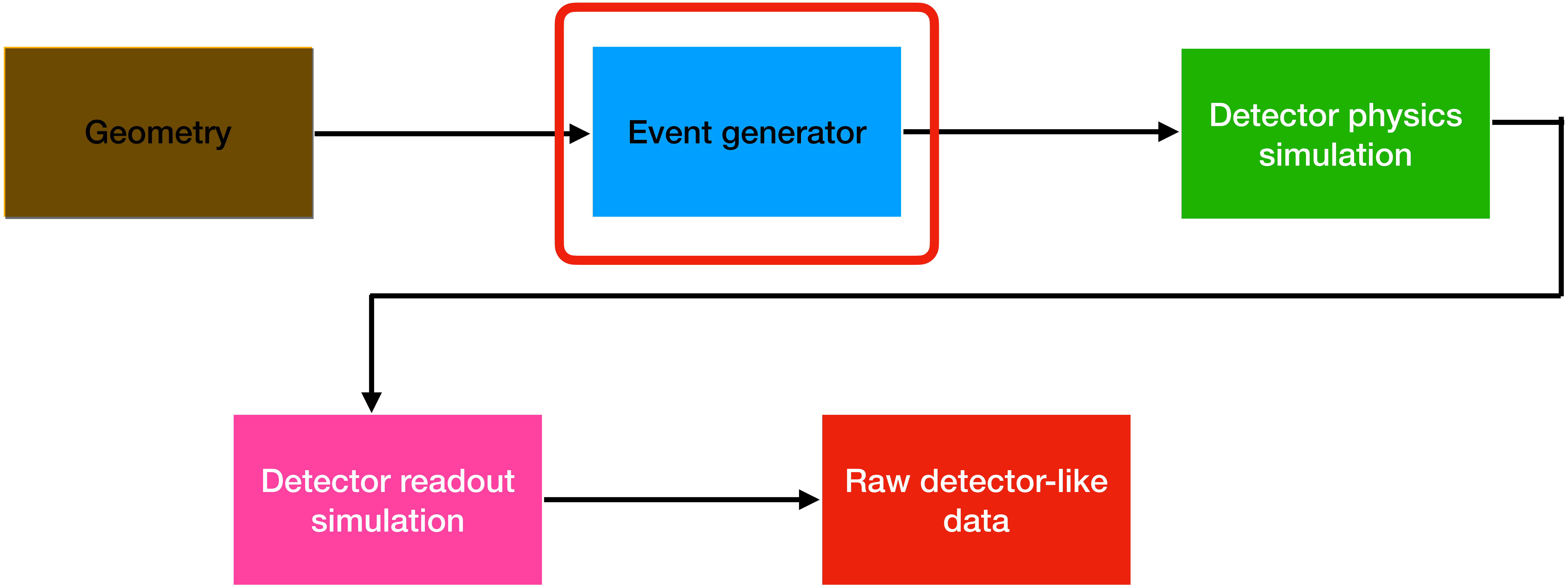
- Use ID objects to specify which instance of TPC geometry objects you want
- There are sorting algorithms in place that determine which one goes first in the code

Coordinate system



For all detectors: Z increases in the direction of neutrino travel, Y increases away from the centre of the Earth and X increases so as to make a right-handed coordinate system.

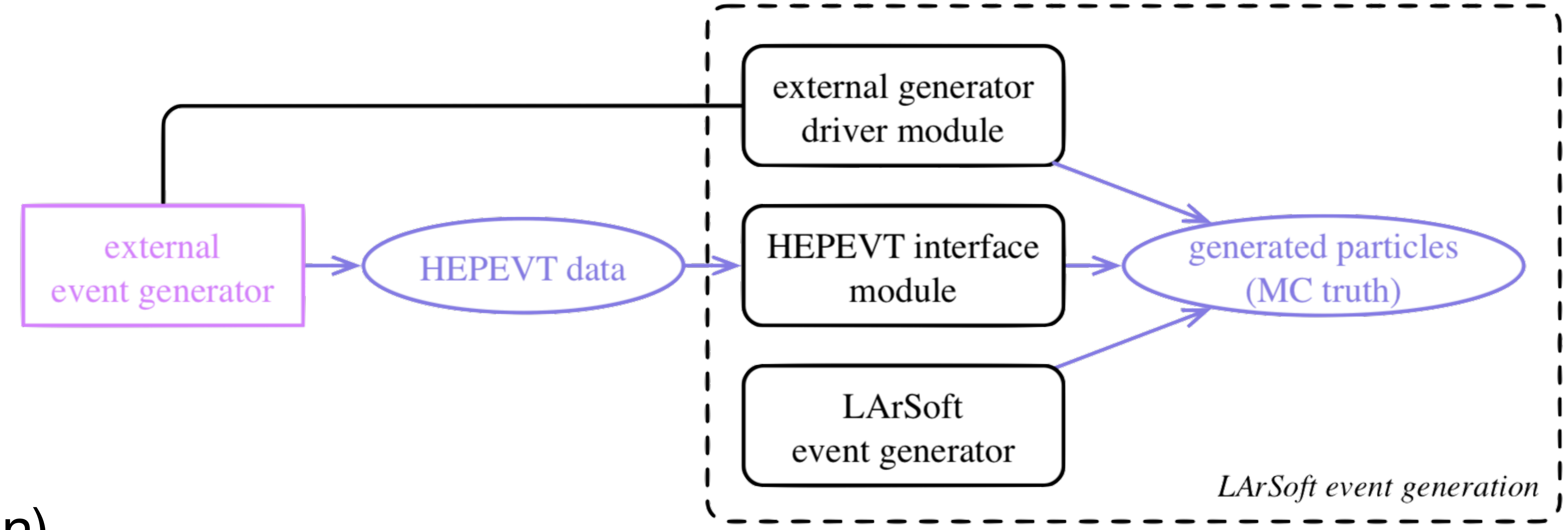
LArSoft simulation flowchart



Event generator(s)

Where we create particles from nothingness

- First step in generating events in LArSoft (majority of cases).
  All generators live in `larsim/EventGenerator`
- We may be interested in different sources of particles:
  - Single particle gun (SingleGen)
  - Neutrino interactions (GENIE)
  - Cosmic rays (CORSIKA)
  - Supernova neutrinos (MARLEY)
  - Read in from text file (TextFileGen)
- Possibility to combine generators to create complex events



Event Generator(s)

Particle Gun: SingleGen

- Single particle gun equivalent in LArSoft. Very useful for debugging code and understanding simple features of what's going on.
- You can define the particle type (PDG code), position, momentum and their how they vary (uniform, gaussian)
- There is an option to run with different/multiple particles either randomly between events or within the same event.
    - This is a bit tricky because you need to specify parameters for all particles. But there is a trick: you can ask LArSoft to "PadOutVectors". Your array then needs to be 1 or N particles (where N is max number)

```
standard_singlep:
{
 module_type:           "SingleGen"
 ParticleSelectionMode: "all"       # 0 = use full list, 1 =  randomly select a single listed particle
 PadOutVectors:     false       # false: require all vectors to be same length
                                    # true:  pad out if a vector is size one
 PDG:                   [ 13 ]      # list of pdg codes for particles to make
 P0:                    [ 6. ]      # central value of momentum for each particle
 SigmaP:                [ 0. ]      # variation about the central value
 PDist:                 "Gaussian"  # 0 - uniform, 1 - gaussian distribution
 X0:                    [ 25. ]     # in cm in world coordinates, ie x = 0 is at the wire plane
                                    # and increases away from the wire plane
 Y0:                    [ 0. ]      # in cm in world coordinates, ie y = 0 is at the center of the TPC
 Z0:                    [ 20. ]     # in cm in world coordinates, ie z = 0 is at the upstream edge of
                                    # the TPC and increases with the beam direction
 T0:                    [ 0. ]      # starting time
 SigmaX:                [ 0. ]      # variation in the starting x position
 SigmaY:                [ 0. ]      # variation in the starting y position
 SigmaZ:                [ 0.0 ]     # variation in the starting z position
 SigmaT:                [ 0.0 ]     # variation in the starting time
 PosDist:               "uniform"   # 0 - uniform, 1 - gaussian
 TDist:                 "uniform"   # 0 - uniform, 1 - gaussian
 Theta0XZ:              [ 0. ]      #angle in XZ plane (degrees)
 Theta0YZ:              [ -3.3 ]    #angle in YZ plane (degrees)
 SigmaThetaXZ:          [ 0. ]      #in degrees
 SigmaThetaYZ:          [ 0. ]      #in degrees
 AngleDist:             "Gaussian"  # 0 - uniform, 1 - gaussian
}

random_singlep: @local::standard_singlep
random_singlep.ParticleSelectionMode: "singleRandom"  #randomly select one particle from the list

argoneut_singlep: @local::standard_singlep

microboone_singlep: @local::standard_singlep
microboone_singlep.Theta0YZ: [ 0.0 ]  # beam is along the z axis.
microboone_singlep.X0:   [125]        # in cm in world coordinates, ie x = 0 is at the wire plane
microboone_singlep.Z0:   [50]         # in cm in world coordinates
```

larsim/EventGenerator/singles.fcl

Event Generator(s)

GENIE

The University of Manchester

- GENIE is the most popular neutrino event generator.
- You provide the flux files and specify where you want the neutrino to interact.
- It produces neutrino secondaries according to flux files appropriate to the detector under study.
- You can specify the type of interaction (CCQE, RES, DIS, etc…).
- GENIE is able to calculate the POT exposure for the generated sample.

larsim/EventGenerator/genie.fcl

```
standard_genie:
{
 module_type:      "GENIEGen"

 DefinedVtxHistRange: false
 VtxPosHistRange:  [0. , 0., 0., 0., 0., 0.] #if DefinedVtxHistRange is set to true VtxPosHistRange sets the hist range of the vertex positio
                                            #It is helpful for dual phase detector for which the range is asymmetric.
 PassEmptySpills:  false
 FluxType:         "mono"     #mono, histogram, ntuple, or simple_flux
 FluxFiles:         ["flugg_L010z185i_neutrino_mode.root"] #name of file with flux histos
 BeamName:         "numi"          #numi or booster at this point - really for bookkeeping
 TopVolume:        "volDetEnclosure"        #volume in which to produce interactions
 EventsPerSpill:   1.              #set != 0 to get n events per spill
 POTPerSpill:      5.e13           #should be obvious
 MonoEnergy:       2.              #in GEV
 BeamCenter:       [-1400., -350., 0.]  #center of the beam in cm relative to detector coordinate origin, in meters for GENIE
 BeamDirection:    [0., 0., 1.]    #all in the z direction
 BeamRadius:       3.              #in meters for GENIE
 SurroundingMass:  0.0             #mass surrounding the detector to use
 GlobalTimeOffset: 10000.          #in ns - 10000 means the spill appears 10 us into the readout window
 RandomTimeOffset: 10000.          #length of spill in ns
 FiducialCut:      "none"          #fiducial cut, see https://cdcvs.fnal.gov/redmine/projects/nusoft/wiki/GENIEHelper
 GenFlavors:       [12,14,-12,-14] #pdg codes of flux generator neutrino flavors
 Environment:      [  ]  # obsolete
 ProductionMode:   "yes"           #turn off the GENIE verbosity
 EventGeneratorList: "Default"
 DetectorLocation: "MINOS-NearDet" #location name for flux window
 MixerConfig:      "none"          #no flux mixing by default
 #MixerConfig:     "swap 12:16 14:16 -12:-16 -14:-16" # example flavor swapping
 MixerBaseline:    0.              #distance from tgt to flux window needs to be set if using histogram flx
 DebugFlags:       0               #no debug flags on by default
 XSecTable: "gxspl-FNALsmall.xml"  #default cross section
}
```

Event Generator(s)

Generate from a file: TextFileGen

The University of Manchester

- To use every time a generator isn't interfaced with LArSoft (#BSM)
- Can generate primary particles from a file containing a list of particles, with PDG code, position, momentum, etc…
- Only takes HEPEVT files as input
- Very simple FHICL file!
- Can be tricky to use…

```
standard_textfilegen:
{
 module_type:    "TextFileGen"
 InputFileName: "input.txt"   #name of file containing events in hepevt format to
                              #put into simb::MCTruth objects for use in LArSoft
}
```

larsim/EventGenerator/textfilegen.fcl

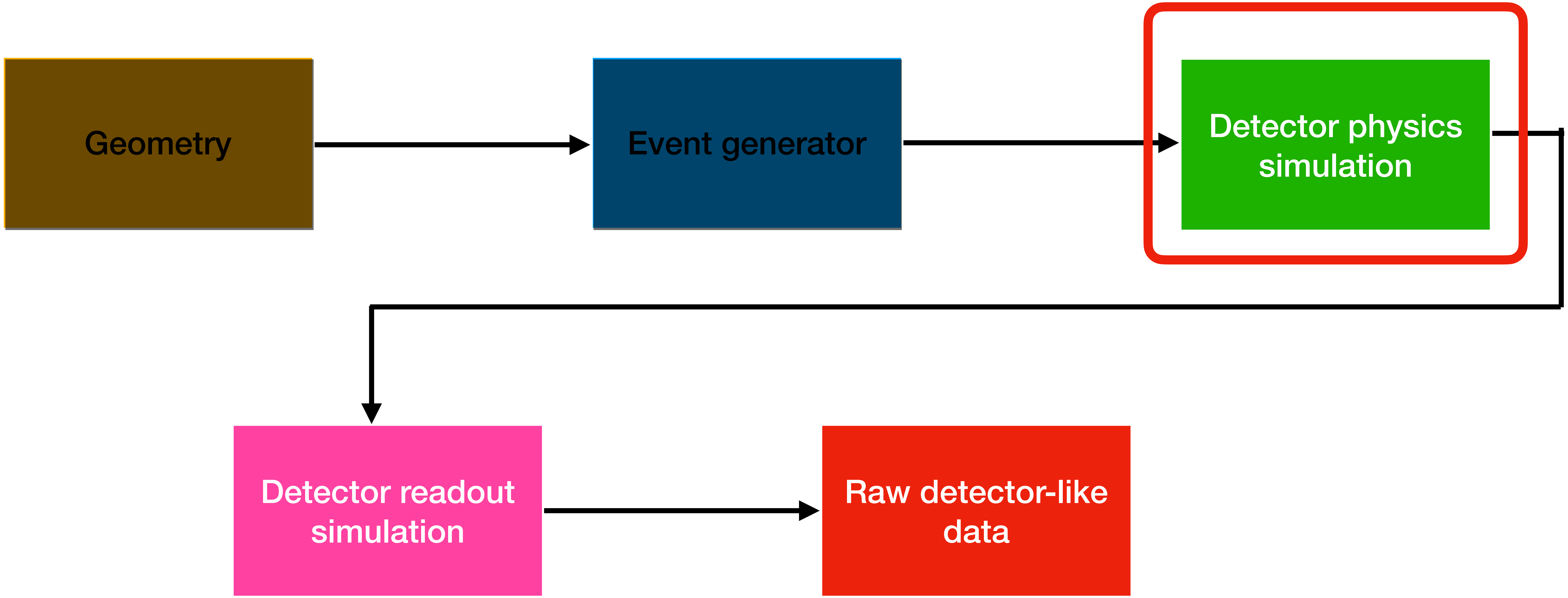
Event Generator(s)

What is in your output file?

The University of Manchester

- `simb::MCTruth` objects (usually one per generator used), which will be picked up by GEANT4 and propagated though the detector.
- Contains:
  - Information about the generator
  - List of particles (`simb:MCParticle`) with PDG code, position, momentum, etc…
  - Information about neutrino interaction (if any)

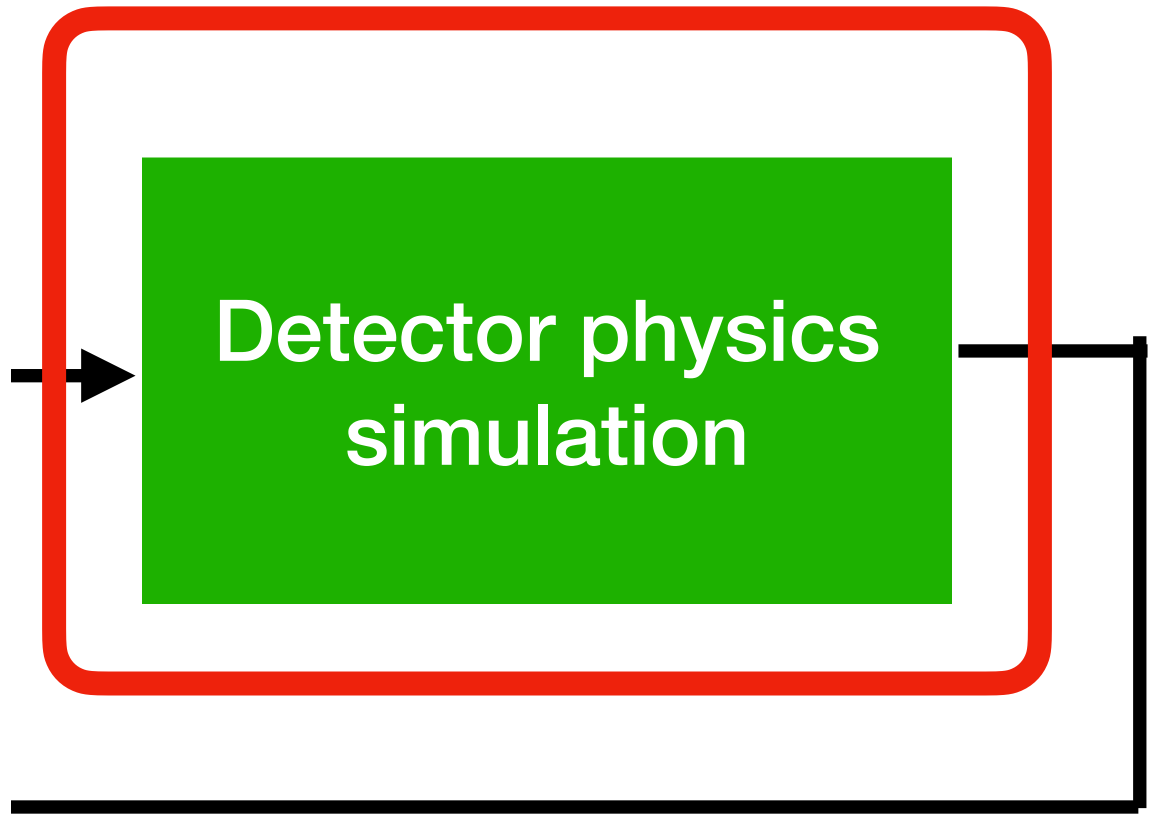
LArSoft simulation flowchart



LArSoft simulation flowchart

MANCHESTER
1824
The University of Manchester

- Interactions of the generated particles with the detector and energy depositions
- Transportation of ionisation electrons and scintillation photons to the readout
- Includes TPC and auxiliary detectors (e.g. CRT)



Parameters for simulation can be found in `larsim/simulation/simulationservices.fcl`

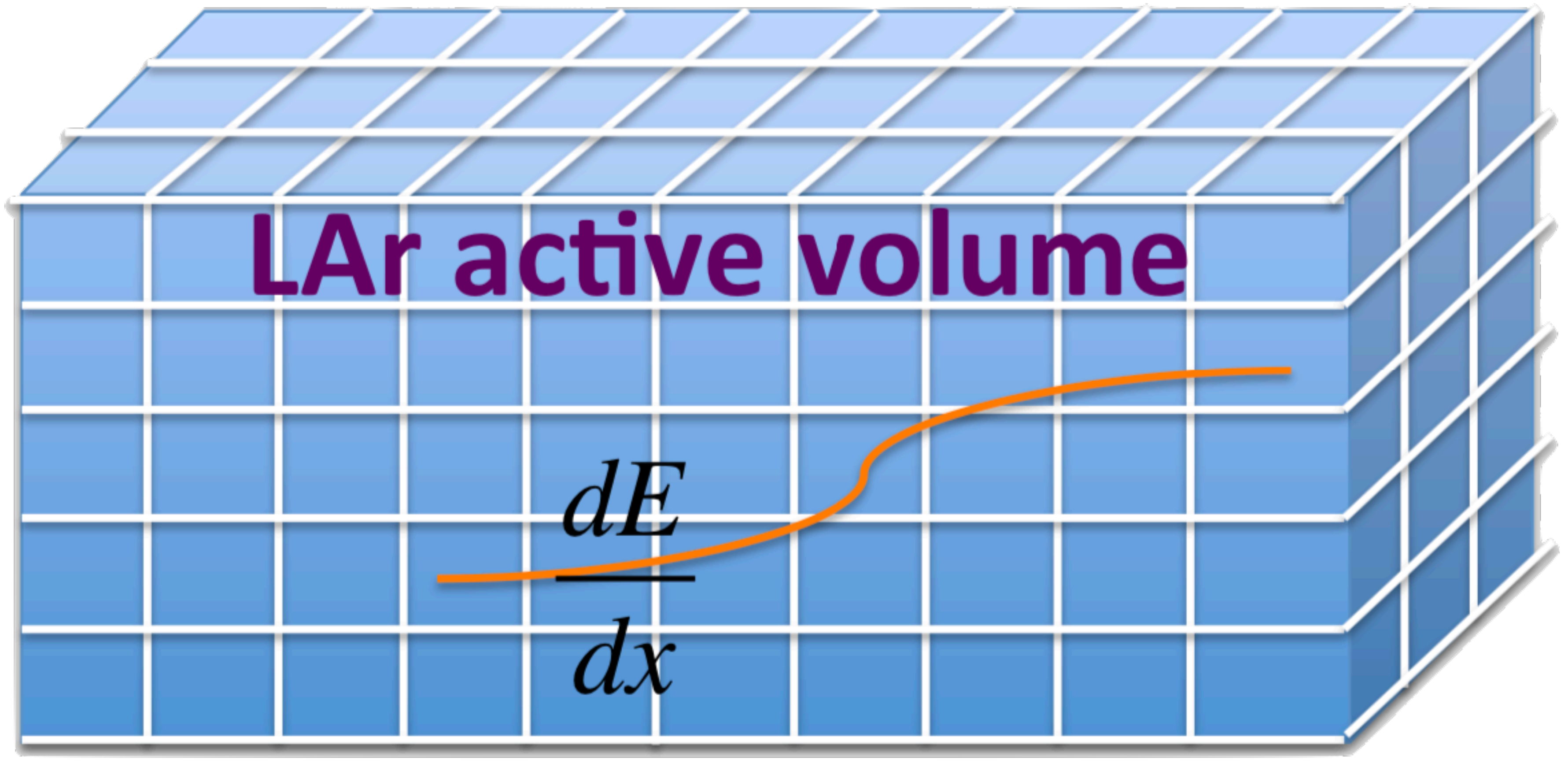
Energy depositions (LArG4)

Where we make our particles interact and see what comes out

- Relies on GEANT4 for particle transportation and energy depositions
- Takes the MCTruth objects from generator stage and passes the primary particles to Geant4 to calculate the energy depositions along propagation though LAr
- Particles are stepped one after the other (oblivious to each other's existence)
  - A step is a 'delta' in the particle trajectory, particle information (energy, position, etc..) is evaluated at each step
  - Step length is calculated based on the physics list (all processes and models to consider for particle interactions)
    - Using QGSP_BERT (recommended one for HEP)

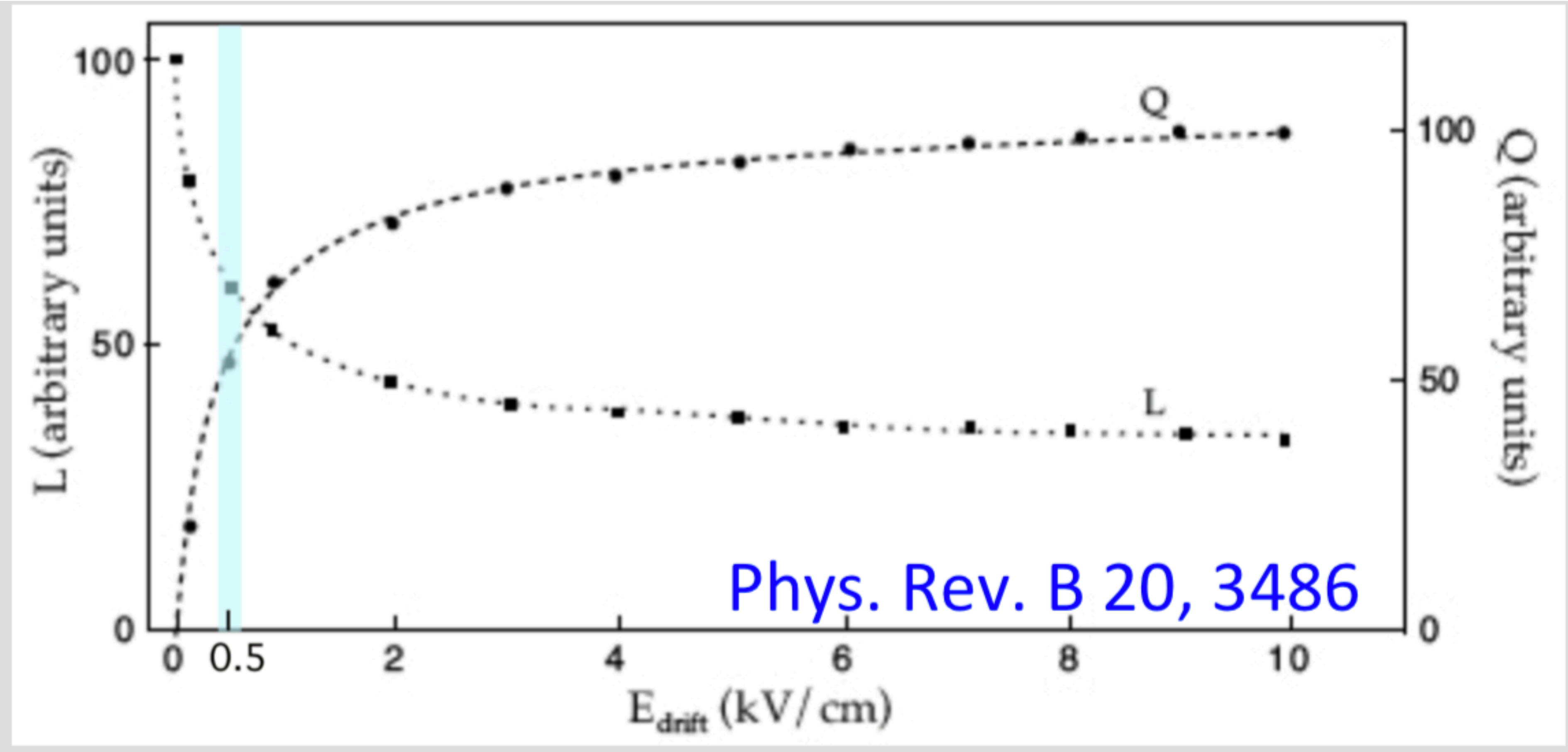
Detector physics simulation (LArG4)

Simulation strategy



- Detector volume divided into voxels (3D pixels)
- Geant4 deposits energy in each voxel

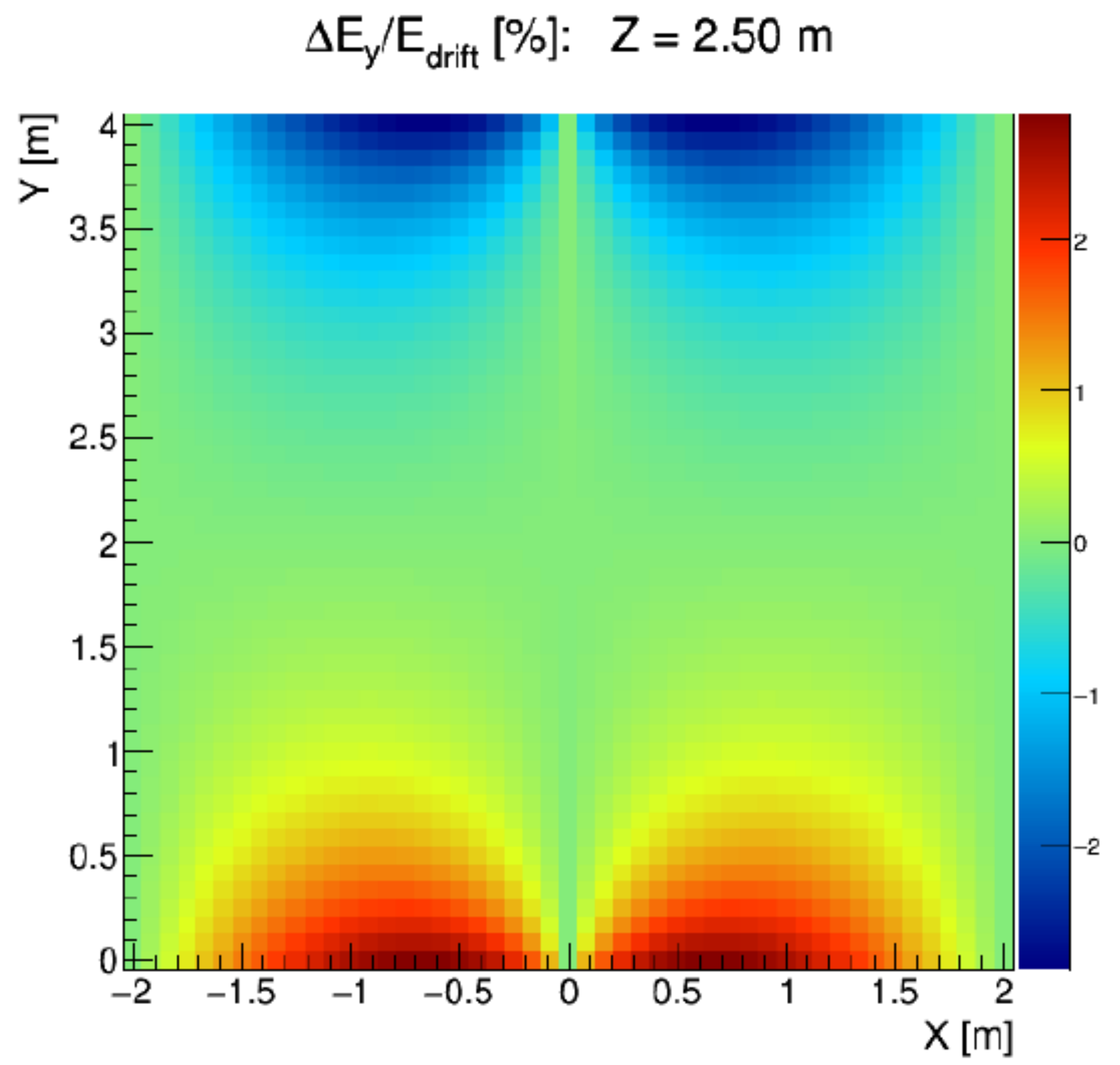
- Number of ionisation electrons and scintillation photons produced depends on the electric field



Detector physics simulation (LArG4)

Electron drift

- Number of ionisation electrons computed from energy deposition
  - dE/dx -> [recombination, lifetime correction (impurities)] -> n_electrons
- Electrons are split in groups (default 600)
- They are projected to a Y, Z position at the position of the wire planes.
- The position is then smeared using transverse diffusion coefficients - this results in an effective diffusion of the whole deposition.
- Longitudinal diffusion is applied the same way
- Generates sequence of arrival times for each channel



Corrections due to field distortions (space charge effect) are applied

Energy depositions (LArG4)

Scintillation photons

MANCHESTER
1824
The University of Manchester

See Andrzej's tutorial on Friday!!

Detector physics simulation (LArG4)

LArG4 is dead! Long live LArG4!

The University of Manchester

There are actually two options for particle propagation: larsim/LArG4 (legacy) and larg4 (refactored).

Legacy

- Based on nutools (general purpose tools for neutrino experiments)
- Obsolete physics lists
- Inefficiencies in interface to Geant4

Refactored

- Based on artg4tk (general art/Geant4 interface)
- GDML extensions
- More recent physics and improved physics list handling
- New implementation of some physical properties

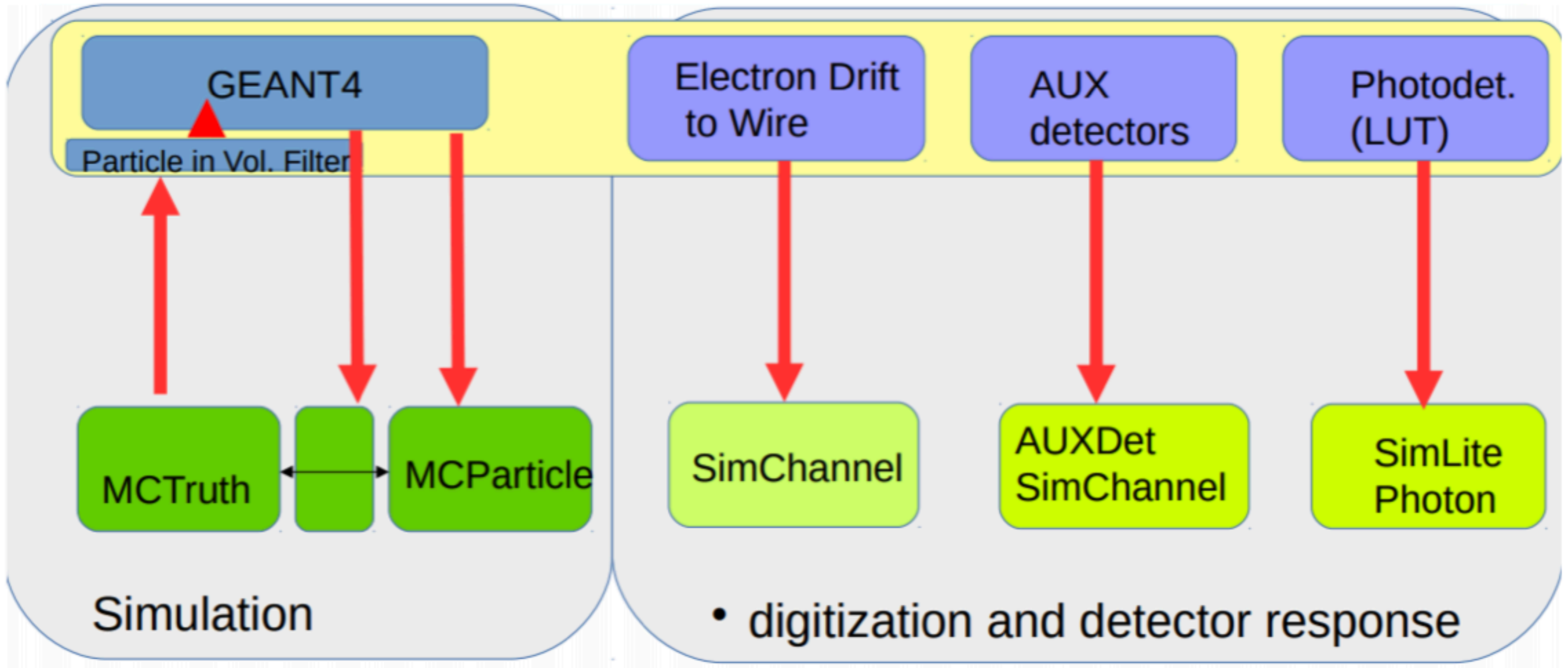
Experiments are migrating to refactored LArG4

Detector physics simulation (LArG4)

LArG4 is dead! Long live LArG4!

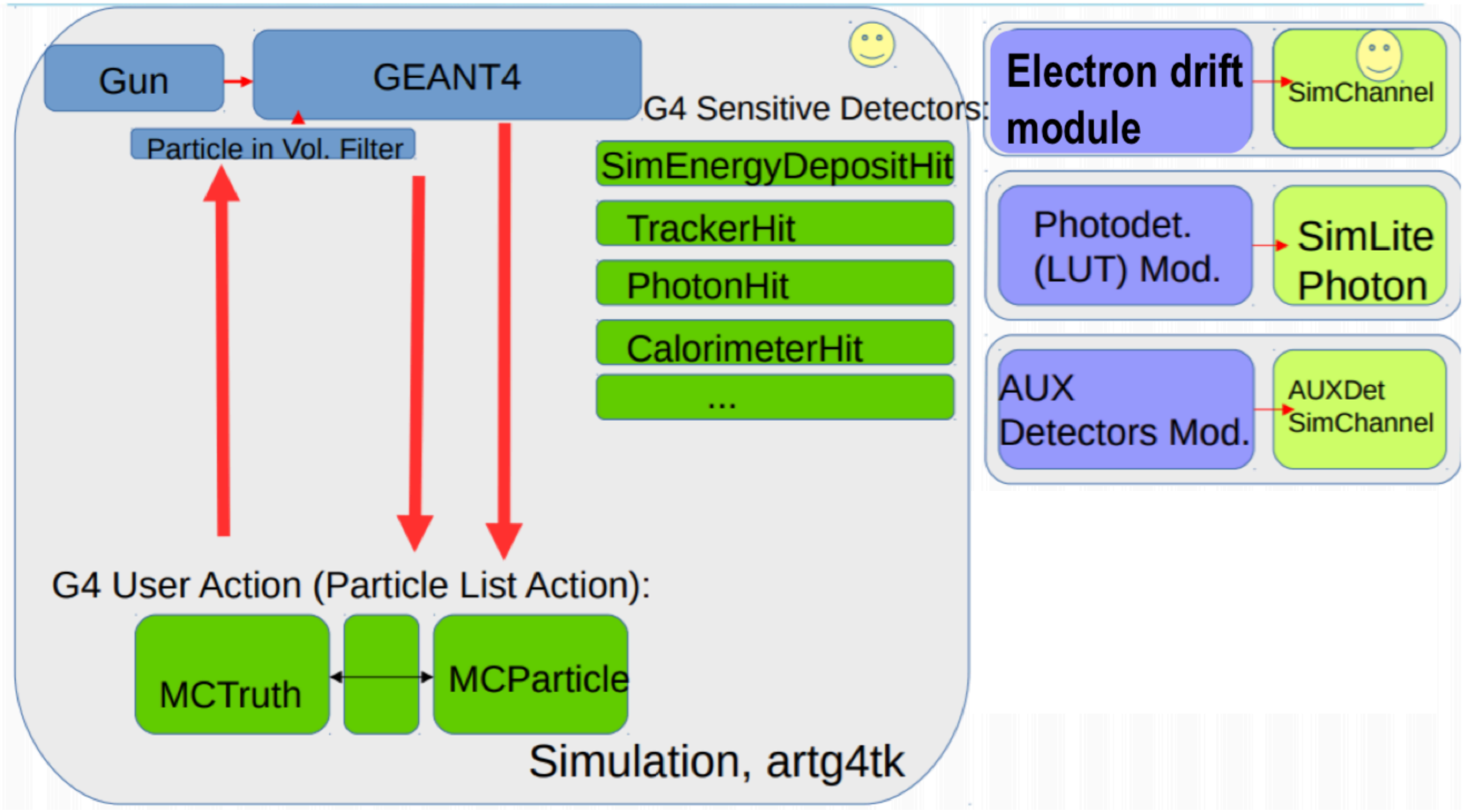
The University of Manchester

Legacy



One module to rule them all

Refactored



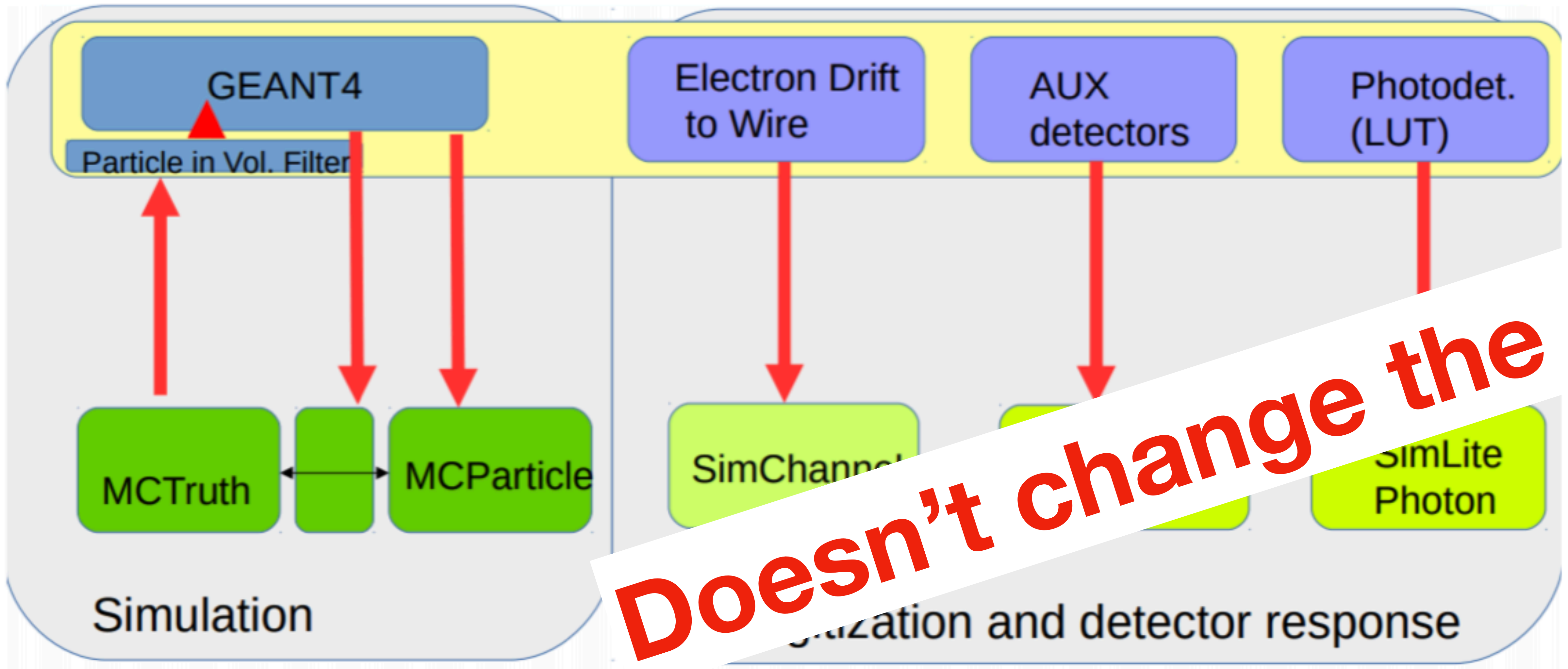
Only simulate particle interaction; separate plugin modules for electron drift, scintillation photons and auxiliary detectors

Detector physics simulation (LArG4)

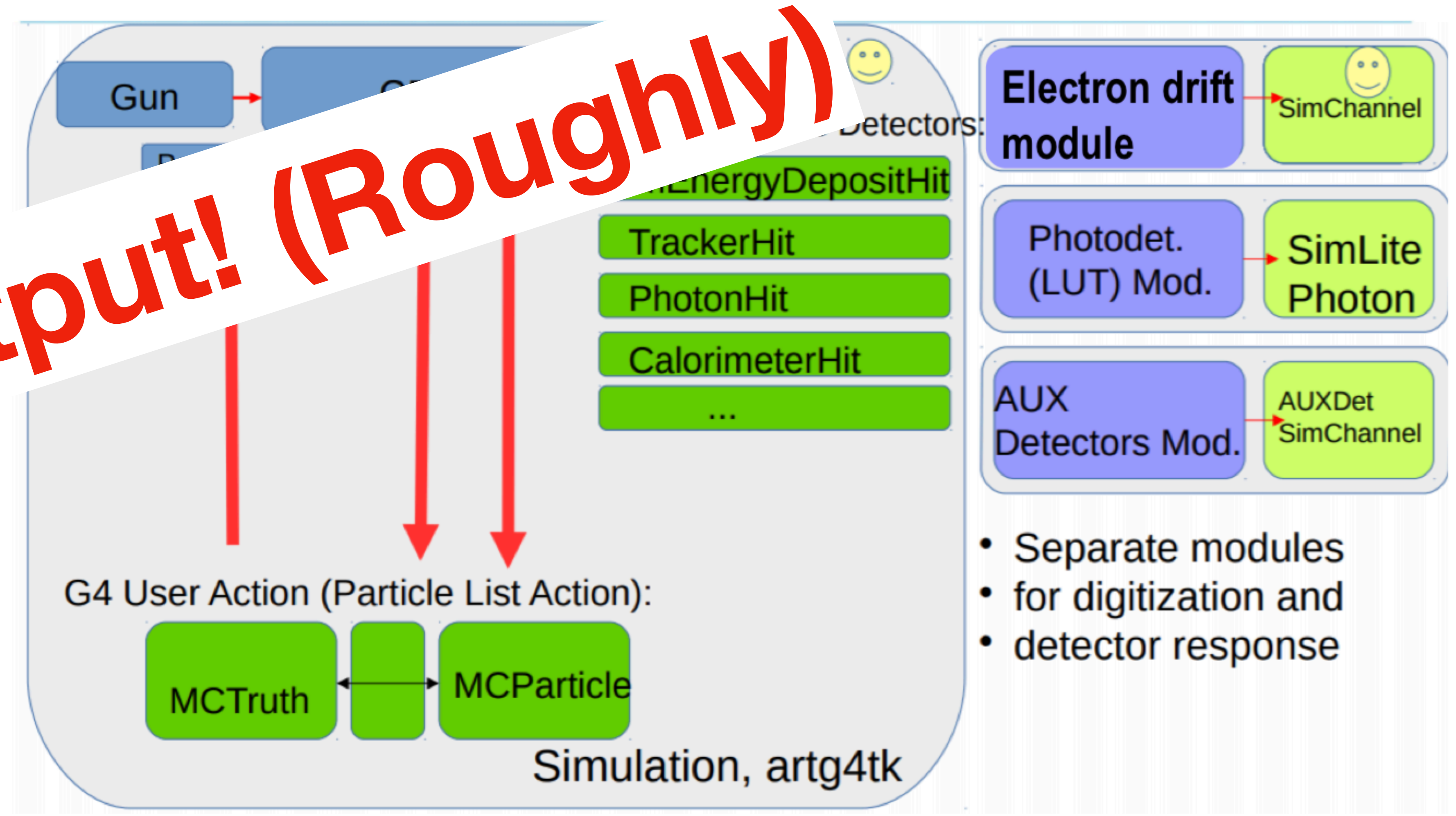
The University of Manchester

LArG4 is dead! Long live LArG4!

Legacy



Refactored



Doesn't change the output! (Roughly)

One module to rule them all

Separate modules for electron drift, scintillation photons and auxiliary detectors

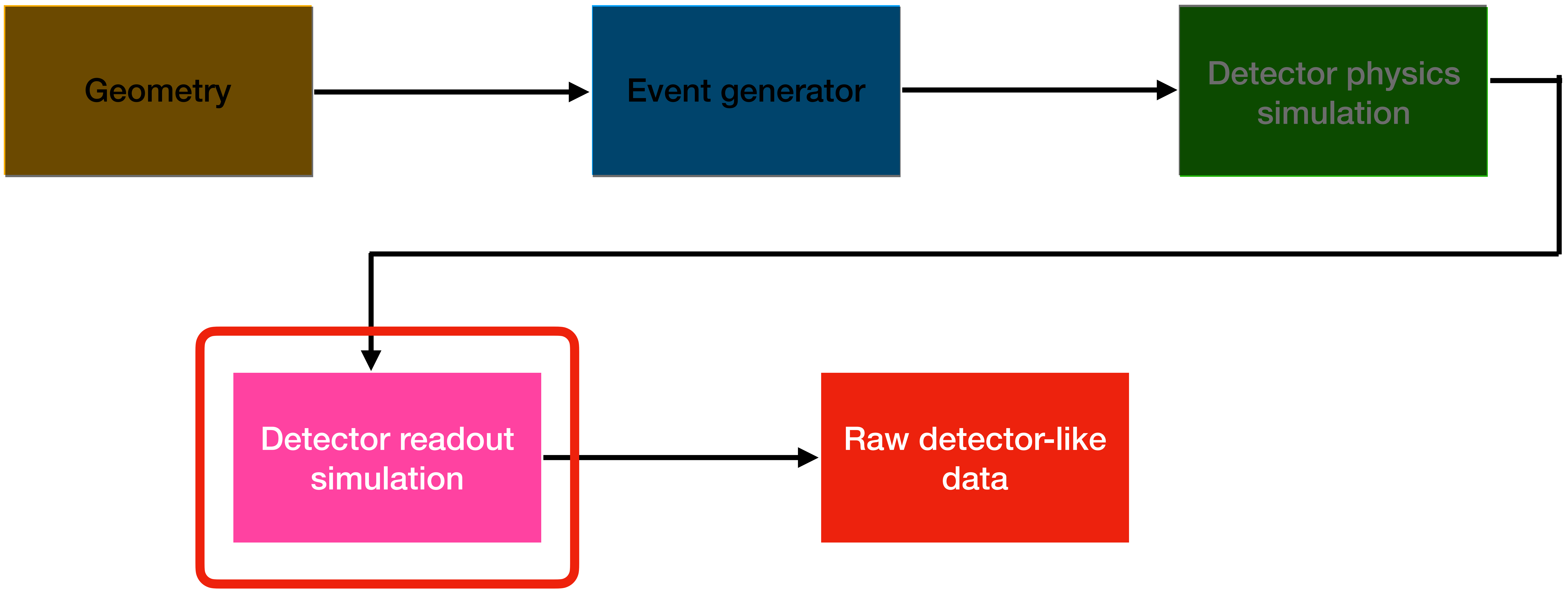
Detector physics simulation (LArG4)

What is in your output file?

- `simb::MCTruth` objects from previous stage.
- New collection of `simb:MCParticle` for particles created during propagation.
- Collections of `sim::SimEnergyDeposit` containing the energy depositions
- Collections of `sim::SimChannel` (wires), `sim::SimPhotons` (optical detectors) and `sim::AuxDetSimChannel` (auxiliary detectors).
    - Contains electrons (photons) reaching the wires (optical detectors) as a function of time, connected to the generated particle that produced them
- With refactored LArG4, you can have more/different data products coming from the plugins.

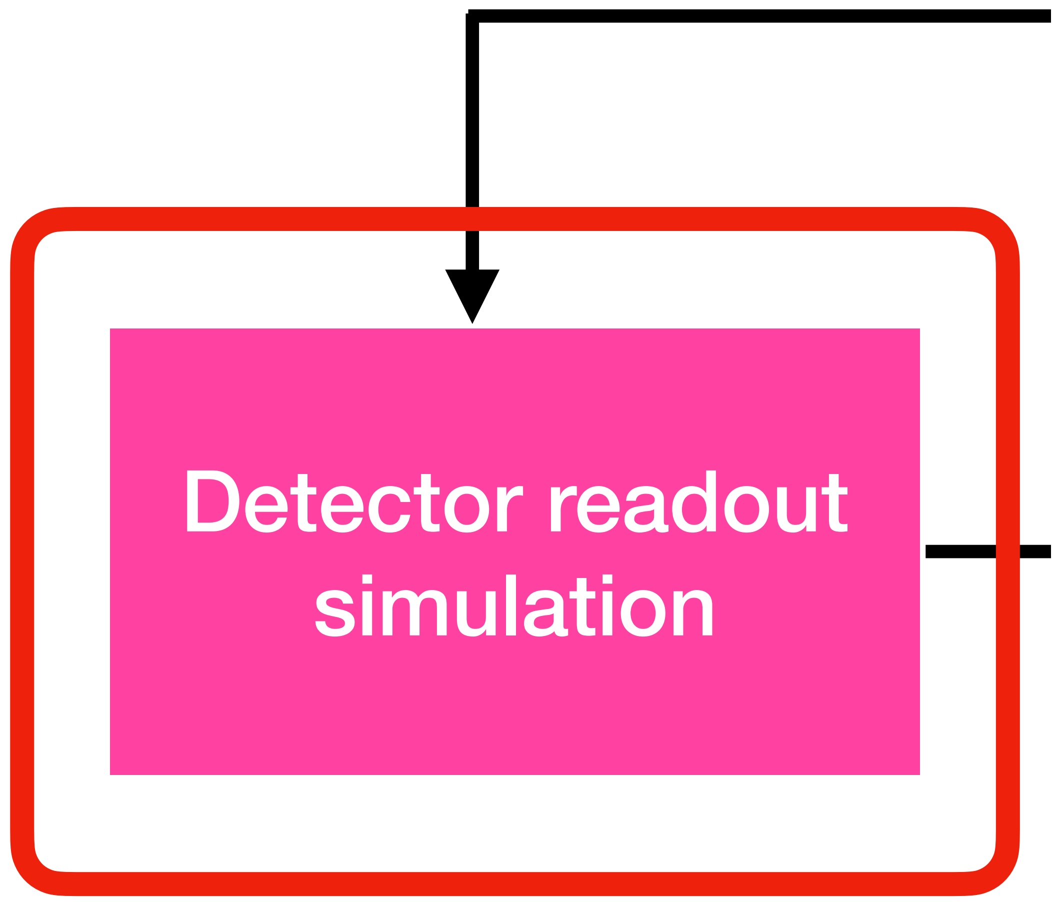
LArSoft simulation flowchart

The University of Manchester



LArSoft simulation flowchart

The University of Manchester



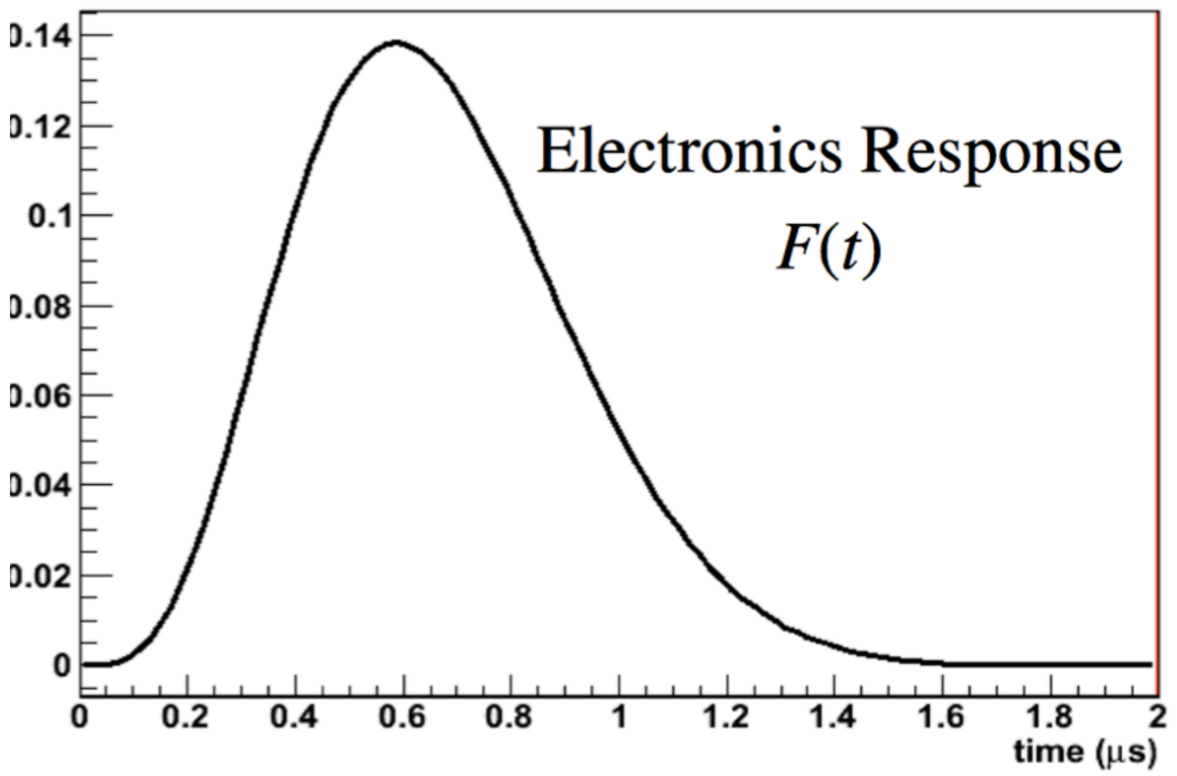
- Transforms the physics information (electrons and photons) into digitised detector response
- Includes the simulation of electronic noise and shaping
- Output is detector-like raw data

Detector readout simulation (DetSim)

Where we make some noise

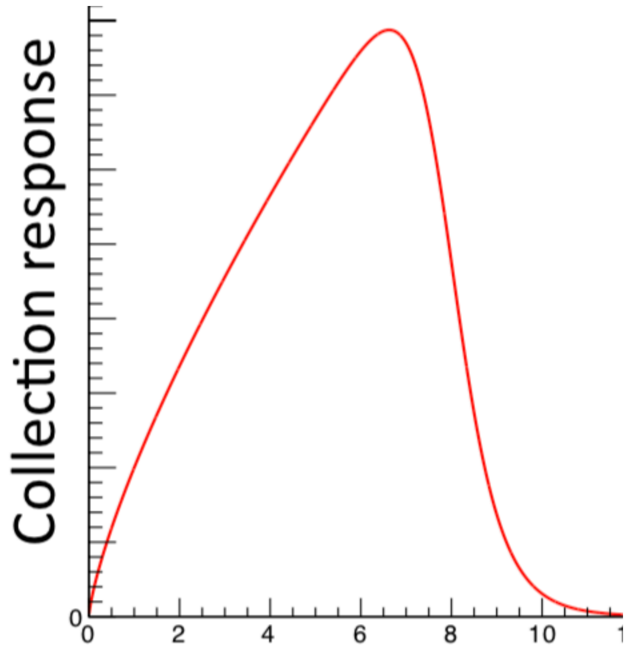
The University of Manchester

Electronics response function ⊗ Field shape + Noise

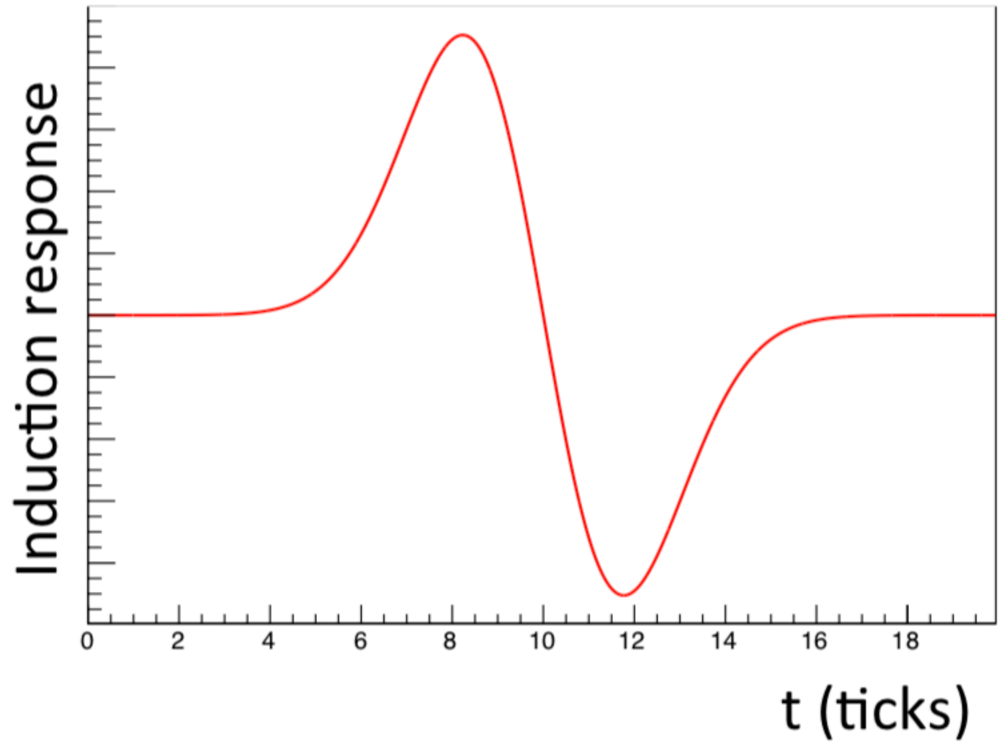


Depends on gain and shaping time

Typical Collection Field response



Typical Induction Field Response



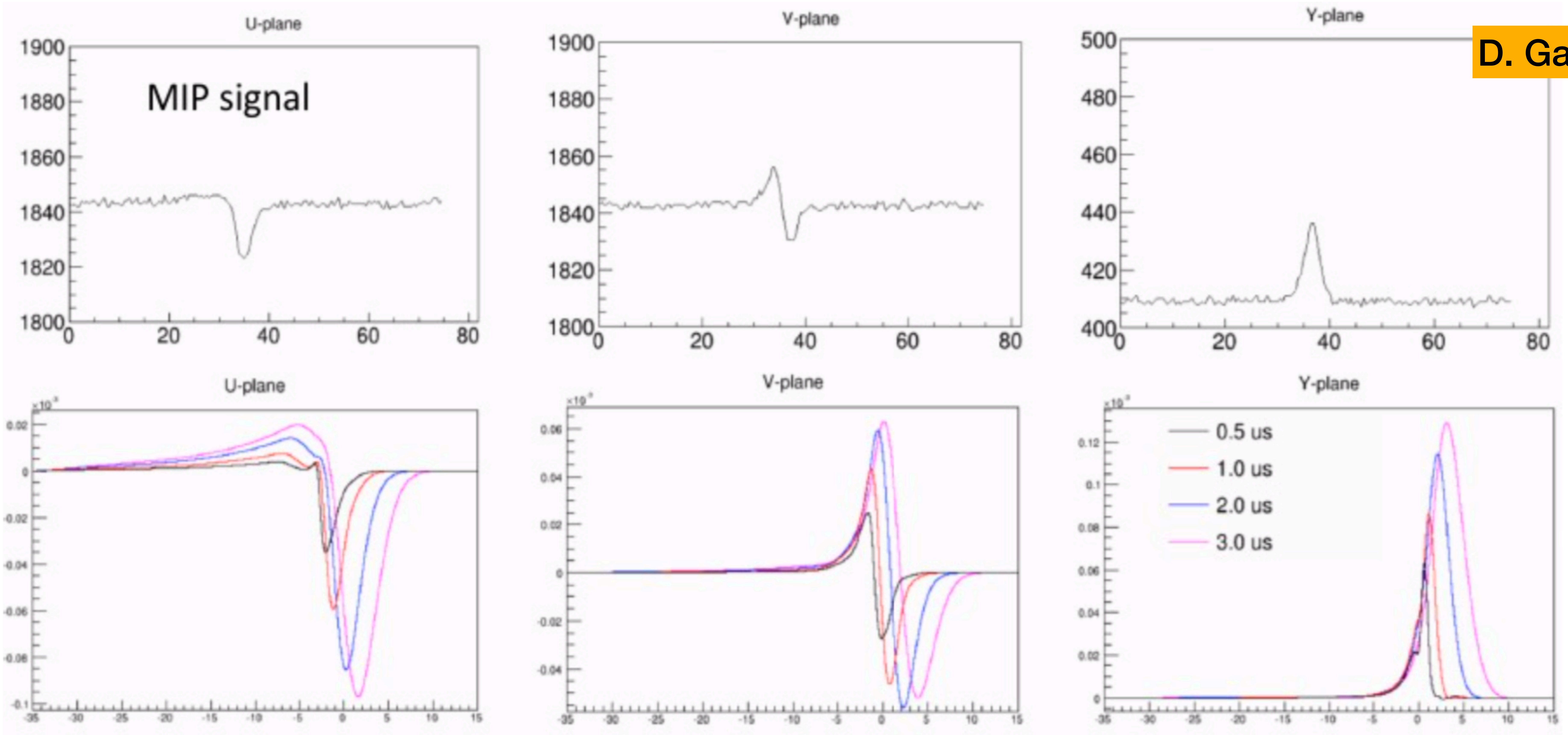
Response to channels to drifting electrons as a function of time

Can be inserted as a histogram (of freq. spectrum), generated in freq. space or with Gaussian distribution in time-domain

Detector readout simulation (DetSim)

Where we make some noise

D. Garcia-Gamez

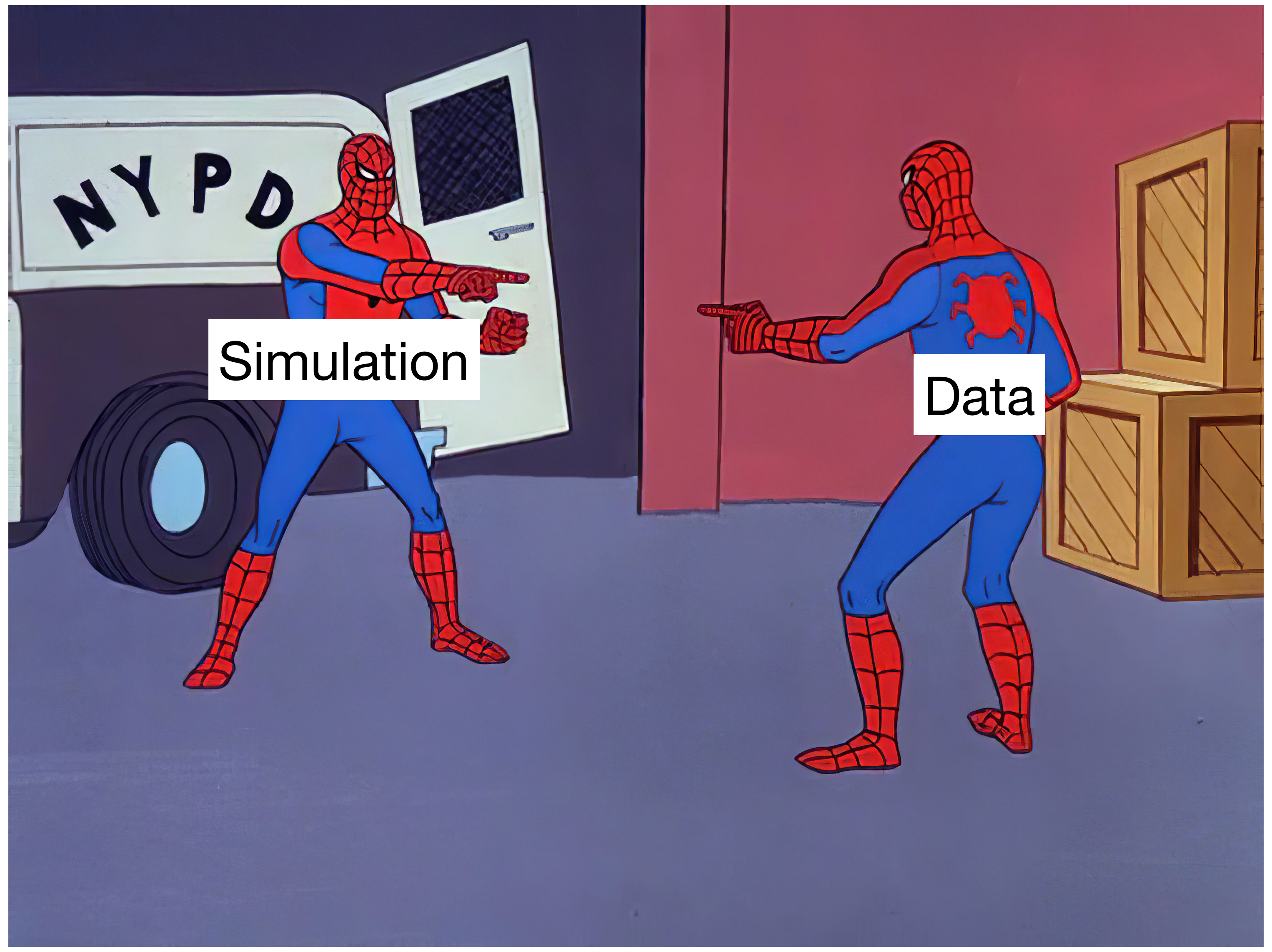


Digitised signal after the ADC = ionisation signal convoluted with the detector and electronics response functions then digitised at a fixed frequency

Detector readout simulation (DetSim)

What is in your output file?

- Objects from the previous stages
- Collection of `raw::RawDigit` and `raw::OpDetWaveform` containing the data-like digitised waveforms



Summary

- Simulation in LArSoft is composed of many steps.
    - It can be scary but you'll learn!
- Offers a lot of possibilities.
- LArSoft is an ever-changing landscape, so you'll have to keep track of new developments.
- Now, let's generate some events!

Sources

Where to find more details

- **LArSoft website:** https://larsoft.org
- **LArSoft wiki:** https://cdcvs.fnal.gov/redmine/projects/larsoft/wiki
- **LArG4 wiki:** https://cdcvs.fnal.gov/redmine/projects/larg4/wiki
- **List and documentation of LArSoft data products:** https://larsoft.org/important-concepts-in-larsoft/data-products
- **Refactored LArG4**: https://indico.fnal.gov/event/18681/contributions/48530/attachments/30244/37222/Dune.pdf
- **Geant4 website:** https://geant4.web.cern.ch

Backup

Deconvolution

Deconvolution

Let's undo everything we just did!

- ✓ So now, that we have the raw data event, we want to remove the field shape and electronics response that we just put in
- ✓ This is done in `CalWire<Experiment>` (again outsourced to `SingalShaping<Experiment>_service`)
- ✓ Filtering can also be applied here to get rid of noise
- ✓ What we want to get is a representation of the initial charge in ADC counts as accurate as possible (without the detector effects) → Which we could then convert to real charge via `DetectorProperties::ElectronsToADC`

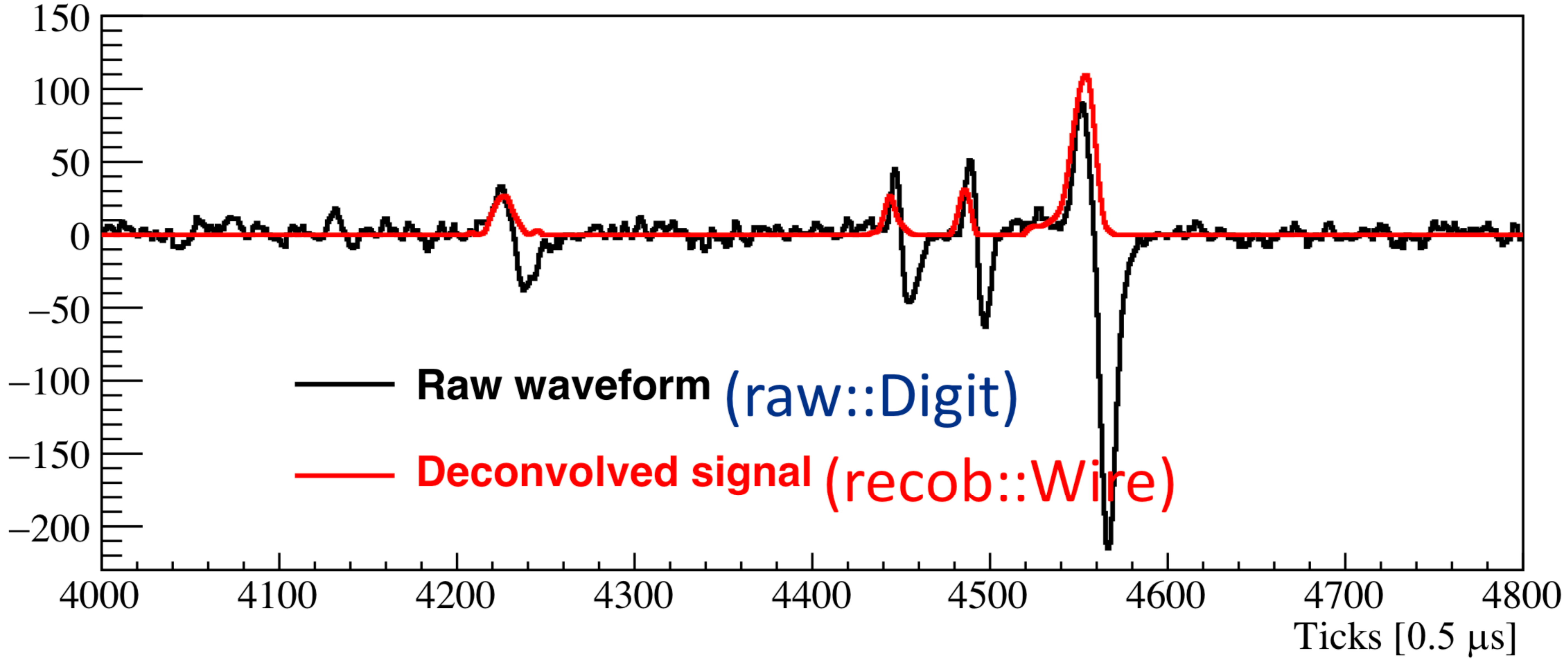
D. Garcia-Gamez

A. Szelc

But wait, there is more!

MANCHESTER
1824
The University of Manchester

- If you can convolve, you can deconvolve
- You obtain a representation of charge vs. time in ADC count



`CalWire` removes field shape and electronics response

Courtesy of Tingjun Yang

Deconvolution

MANCHESTER 1824
The University of Manchester

✓ Deconvolution is needed to convert raw readout signals to arriving charge vs. time
Based on deconvolution kernels (precalculated and stored in a file or calculated on the fly, at job initialization) → `SignalShaping<Experiment>`

✓ **Ideally**, according to image processing theory (Wiener deconvolution), the optimal filter function is (minimizes the mean square error)

$$F(f) = |R(f)|^2 / (|R(f)|^2 + |N(f)|^2)$$

- R(f) = Response function (convolution of field and electronics response)
- N(f) = Noise spectrum

✓ In any case, the deconvolution kernel is calculated as the ratio of the filter function and the convolution kernel

→ **deconvolution kerne: K(f) = F(f)/R(f)**

D. Garcia-Gamez

A. Szelc

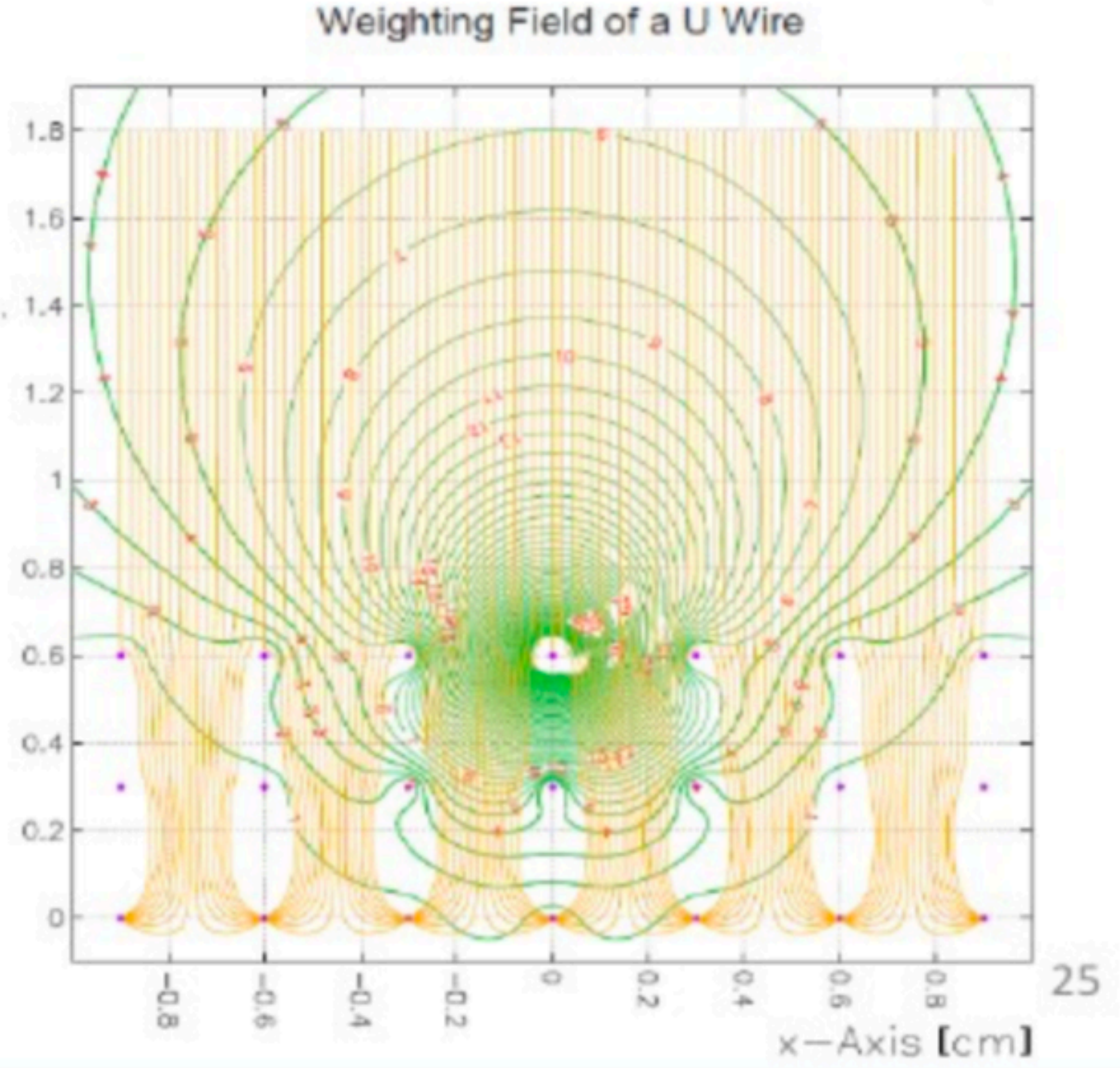
Additional challenges in signal processing

MANCHESTER 1824

The University of Manchester

- With the noise filtering and deconvolution, the full signal processing chain is complete
- But, there are still some additional challenges involved in the process:
    - Ionized electrons traveling through the TPC wires induce signal not only on the closest wire but also on the adjacent wires → dynamic induced charge
    - The field model described before does not take into account the charge contributions from the adjacent wires

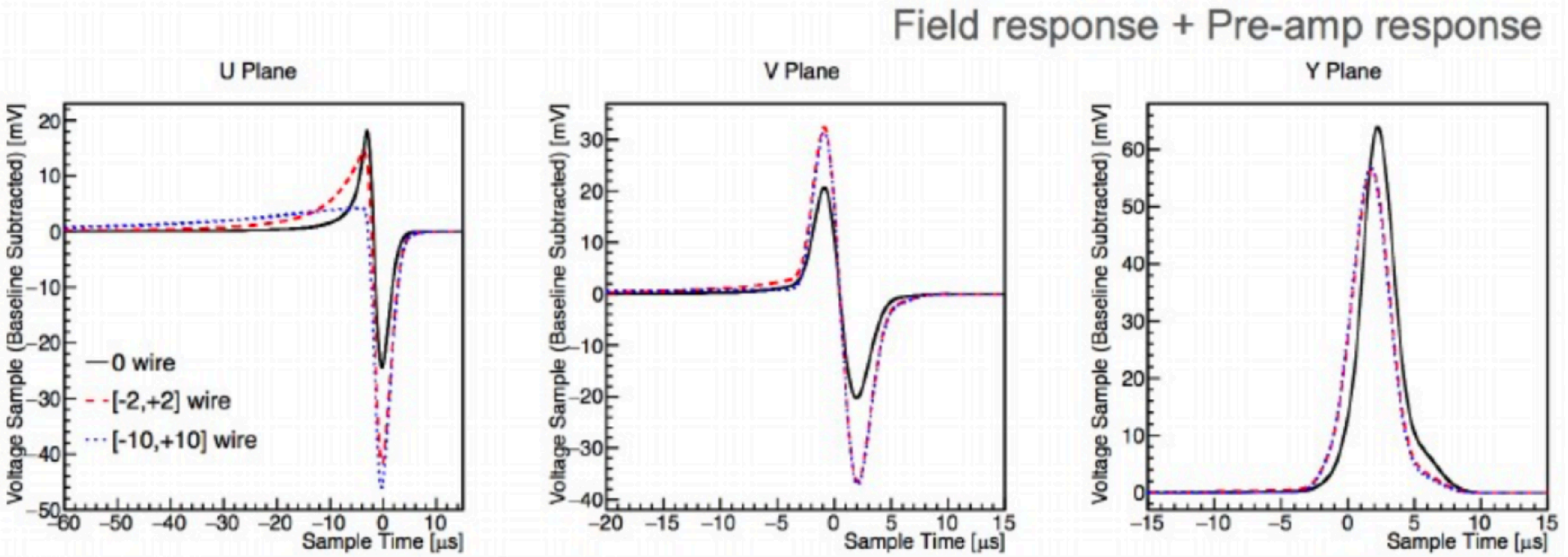
2D GARFIELD simulation : The field extends beyond a single wire region



A. Szelc

Field response - 2D Garfield calculation

MANCHESTER 1824
The University of Manchester



2D Deconvolution

Significant contribution from adjacent wires

- Deconvolve with respect to time and wire dimensions
- $M_i(t') = \int_{-\infty}^{\infty}(\ldots + R_1(t_0 - t) \cdot S_{i-1}(t) + R_0(t_0 - t) \cdot S_i(t) + R_1(t_0 - t) \cdot S_{i+1}(t) + \cdots)dt$
  - $M_i(t')$ - measured signal from wire *i*,
  - $S_i(t)$ - signal within the boundaries of wire *i*, where ± a half pitch defines the wire boundaries
  - $R_n(t_0 - t)$ - average response of wire *i*, where $n = \| i \|$

This implementation of a double de-convolution method is a recent development in the LArSoft signal processing procedure

Courtesy of Brooke Russell

A. Szelc

Event Generators

Event Generator(s)

HEPEVT format

MANCHESTER 1824
The University of Manchester

Each event is described by at leat two lines:
- First is event number and number of particles (ignored by art/LArSoft)
- Each describes one particle and contains 15 entries

1. status code (should be set to 1 for any particle to be tracked, others won't be tracked)
2. the pdg code for the particle
3. the entry of the first mother for this particle in the event, 0 means no mother
4. the entry of the second mother for this particle in the event, 0 means no mother
5. the entry of the first daughter for this particle in the event, 0 means no daughter
6. the entry of the second daughter for this particle in the event, 0 means no daughter
7. x component of the particle momentum
8. y component of the particle momentum
9. z component of the particle momentum
10. energy of the particle
11. mass of the particle
12. x position of the particle initial position
13. y position of the particle initial position
14. z position of the particle initial position
15. time of the particle production

For example, a single muon with a 5 GeV energy moving only in the z direction will be described by:

```
0 1
1 13 0 0 0 0 0. 0. 1.0 5.0011 0.105 1.0 1.0 1.0 0.0
```

LArG4

Detector physics simulation (LArG4)

LArG4 is dead! Long live LArG4!

The University of Manchester

GDML description of the detector

- Materials, volumes, optical properties, …
- Make use of formulas and loops
- Assignment of optical surfaces
- Assignment of sensitive detectors of predefined type to logical volumes→ automatically trigger the creation and filling of the appropriate hit collections

Energy depositions (LArG4)

Auxiliary detectors

The University of Manchester

- Geometry specifies N AuxSensitiveGeo per AuxDetGeo
- GEANT4 deposits energy in each `AuxDetSensitiveGeo`
- LArG4 stores information about energy deposition in `sim::AuxDetSimChannel`

Energy depositions (LArG4)

MCHit and MCreco

MANCHESTER 1824
The University of Manchester

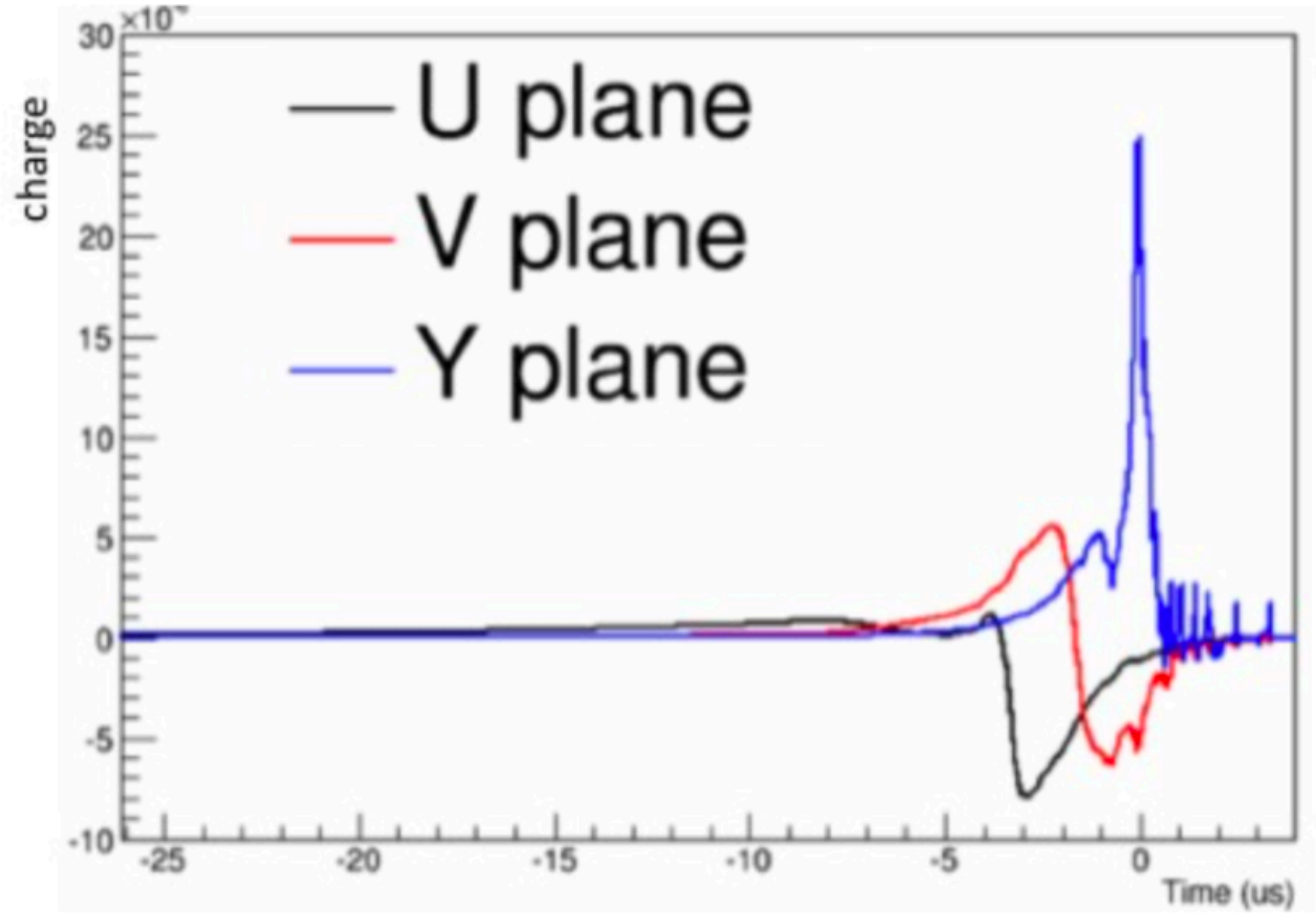
- MCHit is the charge from a single particle seen by a TPC readout channel
- Produces special “true” track and shower objects to evaluate reconstruction algorithms performance

DetSim

Detector readout simulation (DetSim)

Field modelling

- Simulating the field response function is the first step in the chain of signal processing
- The response of the channels to the drifting electrons is parameterized as a function of drift time, with separate response functions for collection and induction wires



- 2-D GARFIELD(*) simulated response to a single electron generated in the MicroBooNE detector
- Can be inserted via Tformula, or use basic step functions

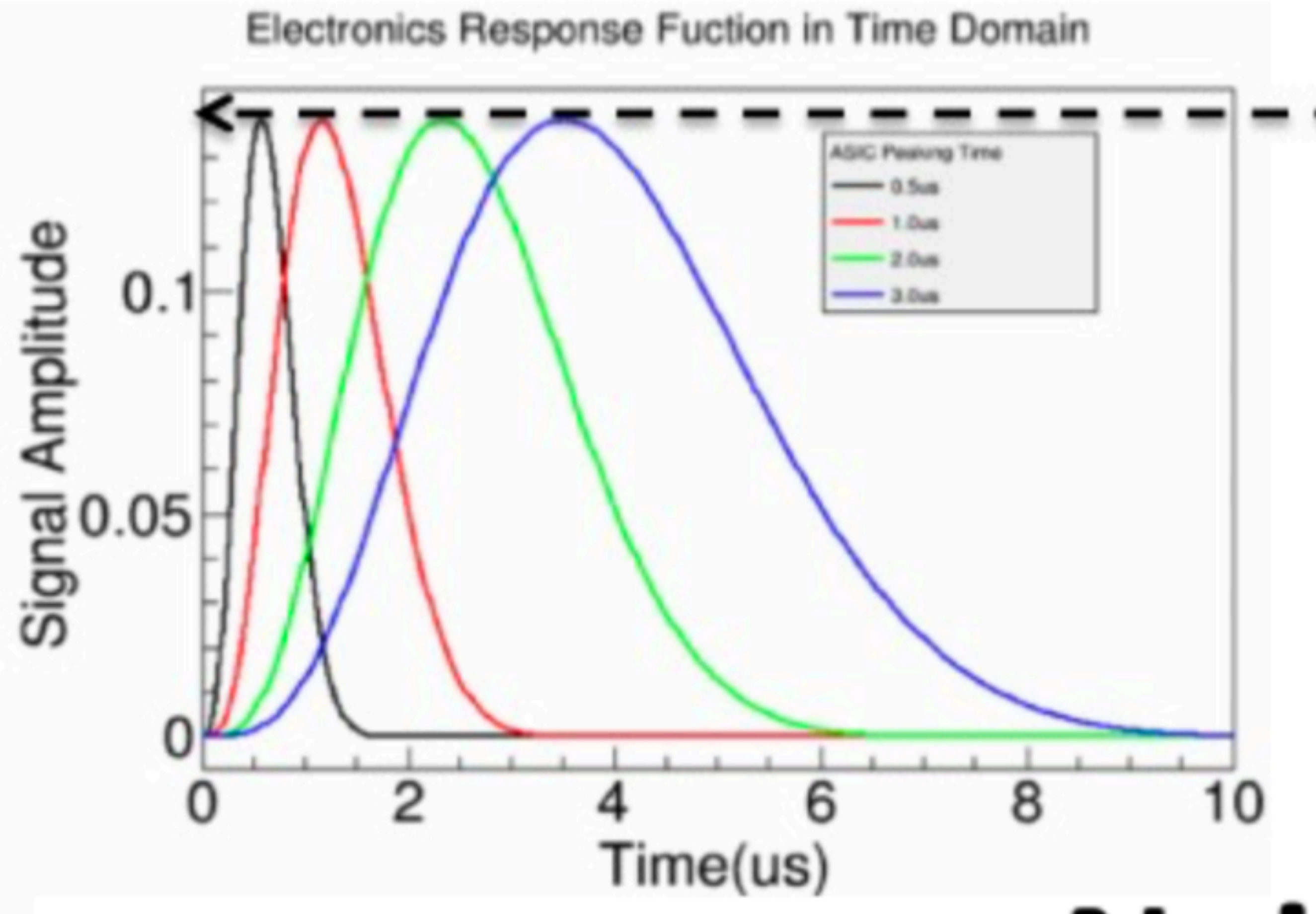
D. Garcia-Gamez

A. Szelc

Detector readout simulation (DetSim)

Electronics response modelling

- MicroBooNE front-end cold electronics designed to be programmable with 4 different gain settings (4.7, 7.8, 14, and 25 mV/fC) and 4 shaping time settings (0.5, 1, 2, and 3 us) → the electronic response function varies according to these settings



- For a fixed gain setting, the peak is always at the same height independent of the shaping time

Noise

- Can be inserted as a histogram (of freq. spectrum), generated in freq. space or with Gaussian distribution in time-domain

D. Garcia-Gamez

A. Szelc