

DUNE FD sim/reco meeting
23/11/2015

HighLAND

a potential analysis framework for DUNE

Anselmo Cervera, J.J. Gomez-Cadenas, A. Izmaylov, P. Novella, M. Sorel

IFIC-Valencia

Introduction

- **HighLAND**, **High Level Analysis** at the **Near Detector**, is the T2K ND analysis framework
- In **DUNE** we would have to change: **Neutrino**
 - or **High Level Analysis** in **DUNE**
- **Highly optimized, thread safe, compiled c++ code** and run on the shell command line (not as root macro)
- It is **extensible** meaning that users commit their analyses into the framework such that other users can reuse common tools

What HighLAND provides

● General analysis tools

- Event loop
- Tools for multiple simultaneous event selections
- Tools for numerical systematic error propagation

● Tools for drawing the analysis results

● Data Reduction functionality. Example:

- LArSoft --> MiniTree --> MicroTree --> NanoTree

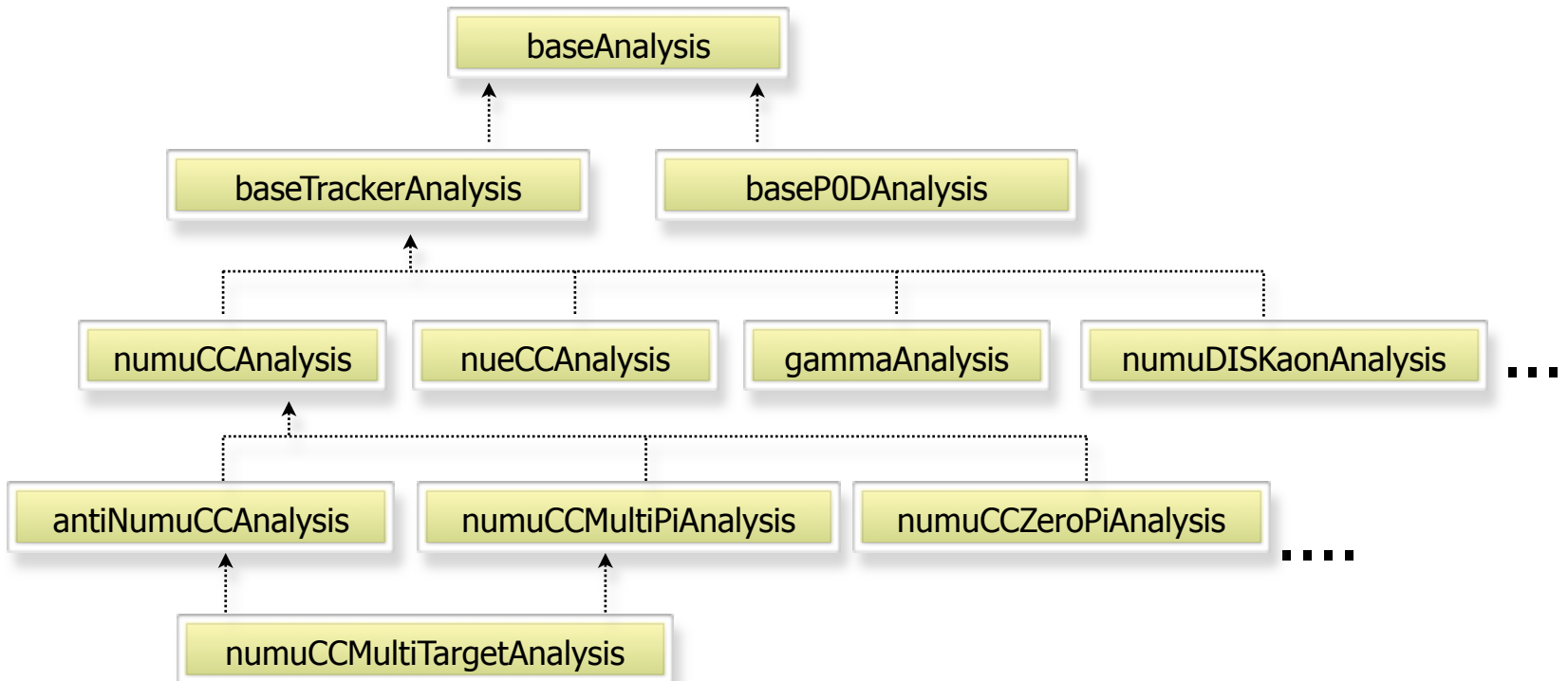
● Tools for incorporating specific analyses into the framework

- Extensible event data model
- Hierarchy of analyses depending on each other

A hierarchy of analyses

Analysis hierarchy in T2K ND

only few
packages
shown here
(>20)



- In most cases packages down in the hierarchy perform selections that are subsamples of the packages above
- But this is not mandatory, you can just use functionality from another package

Why HighLAND and why now?

- In T2K we moved into **HighLAND** only 3 years ago, 2 years after the start of the data taking
- Previously there were several frameworks hanging around
 - Information exchange was complicated
 - Comparing analyses almost impossible
 - A nightmare for newcomers
- Now we have many analyses using the same tools, and what is very important, reusing functionality from other analyses (common cuts, utility methods, etc)
- **Learning curve has decreased considerably**

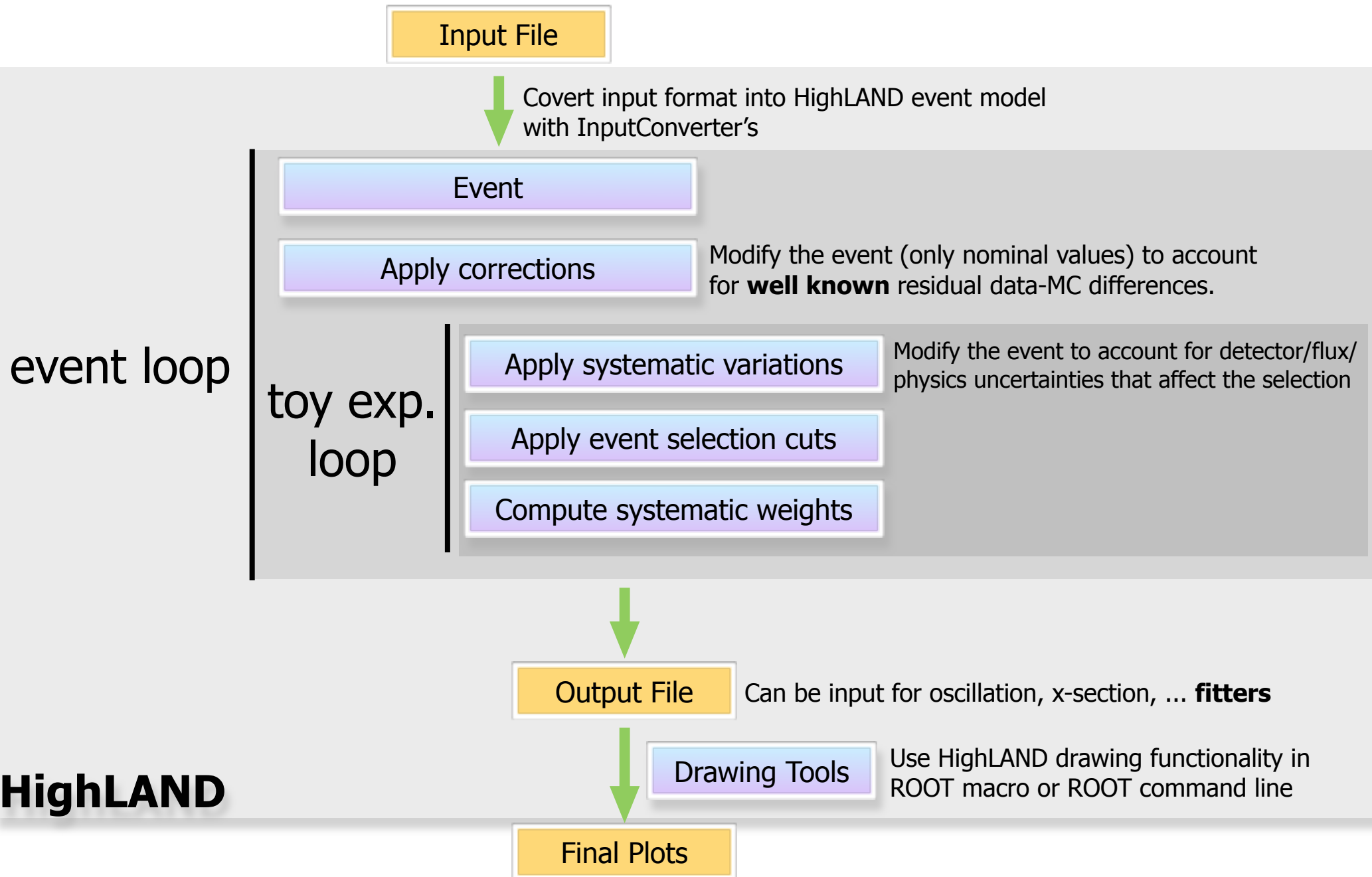
HighLAND for DUNE

- Having a common framework from the very beginning can be very beneficial as we will see later
- Ideally the same framework for all project components:
 - **FD, ND, prototypes**: optimization, input for oscillation analyses, x-section, proton decay, other new physics, etc.
- This is possible because:
 - **HighLAND** can accept **any input format**
 - The basic **event model can be extended** by the user to match the requirements of its particular analysis

Benefits of common framework

- We can imagine all subgroups using the same tools for performing event selection, propagating systematics and doing final plots:
 - In this scenario moving from one group to another should be easier
 - Useful for understanding common systematics between near and far detector
 - People from different groups would speak the same language when talking about selections, systematics and their associated technicalities

Analysis flow



Analysis flow

Input File



Covert input format into HighLAND event model with InputConverter's

Event

Apply corrections

Modify the event (only nominal values) to account for **well known** residual data-MC differences.

Apply systematic variations

Modify the event to account for detector/flux/physics uncertainties that affect the selection

Apply event selection cuts

Compute systematic weights

event loop

toy exp.
loop

Output File

Can be input for oscillation, x-section, ... **fitters**

Drawing Tools

Use HighLAND drawing functionality in ROOT macro or ROOT command line

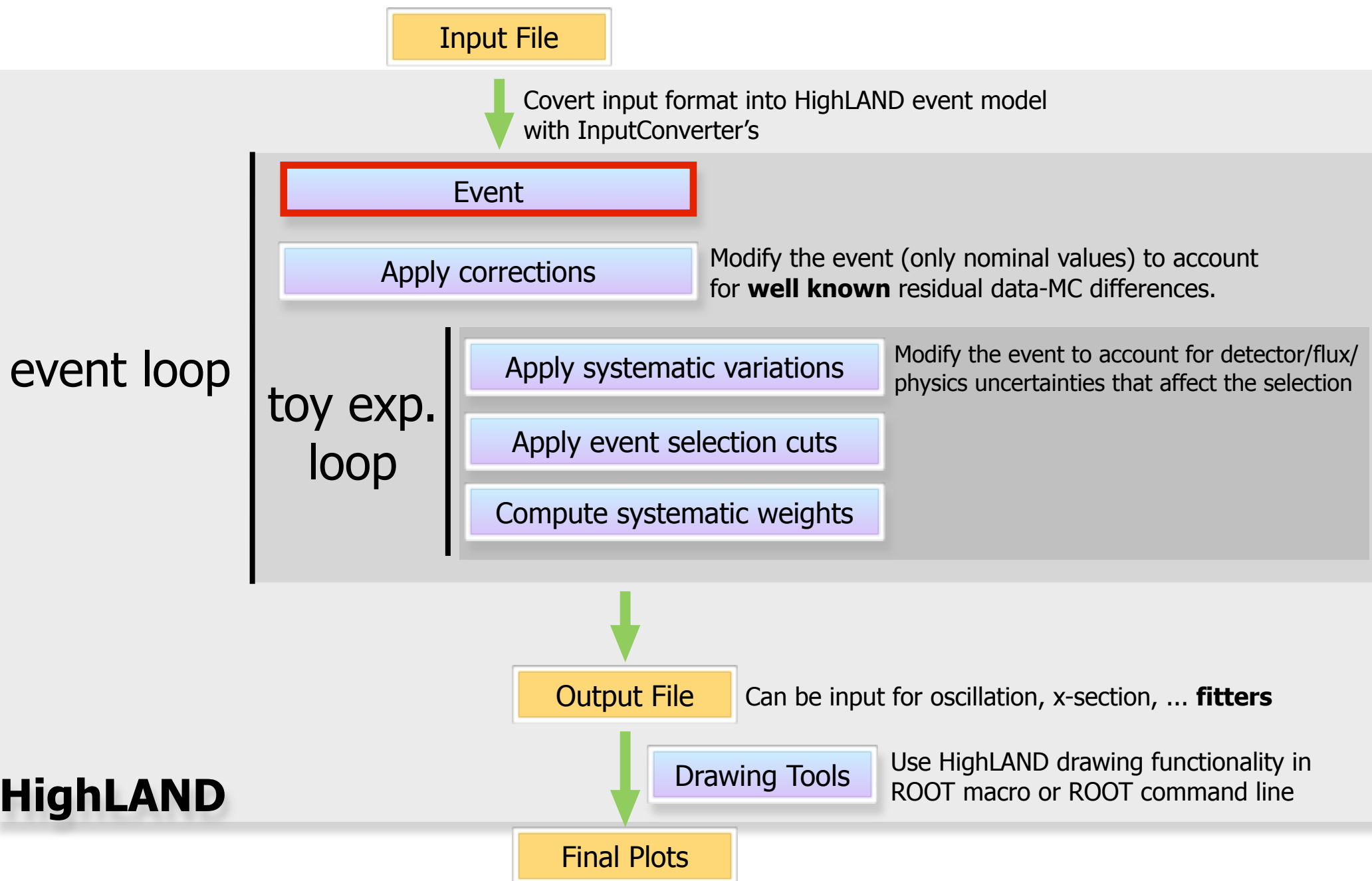
Final Plots

HighLAND

Input Data

- The Input data for **HighLAND** can have any format (in T2K we use root files)
 - For **DUNE** either **Art event** or **AnalysisTree** or ...
- The input file information is dumped into the HighLAND data classes (event model) by **InputConverter's**, one for each input file type
- Once the information is propagated to those data classes, all analyses are independent of the input format
- Input files should be as small as possible to gain in speed and portability
 - **HighLAND** provides a new level in **data reduction** as we will see later

Analysis flow

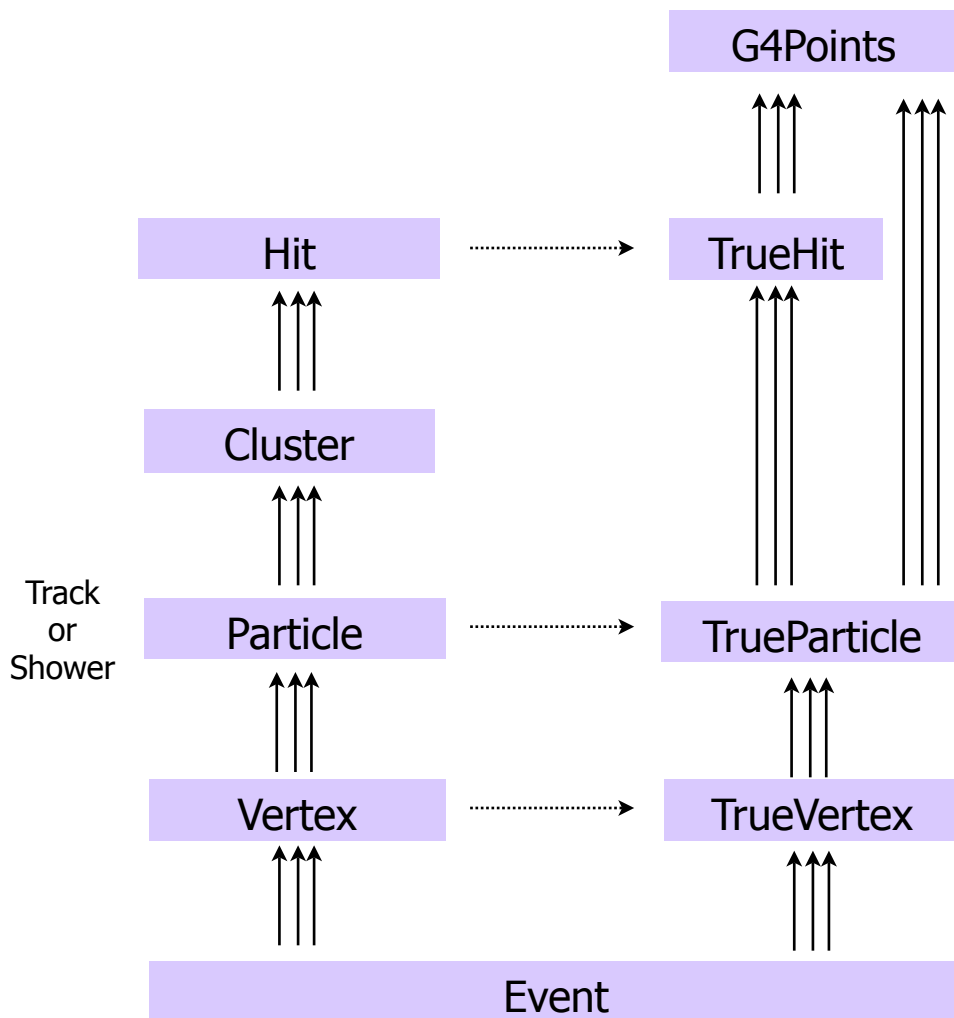


Event Model

- There is a set of basic Data Classes common to all inputs that define the HighLAND event model
- But we can have an extensible data structure inheriting from the base one
 - Various types of analyses needing more complex objects
 - In the same detector, in different detectors, ...
 - etc
- Conversion from base to derived classes should be done with **static_cast** (it's fast)

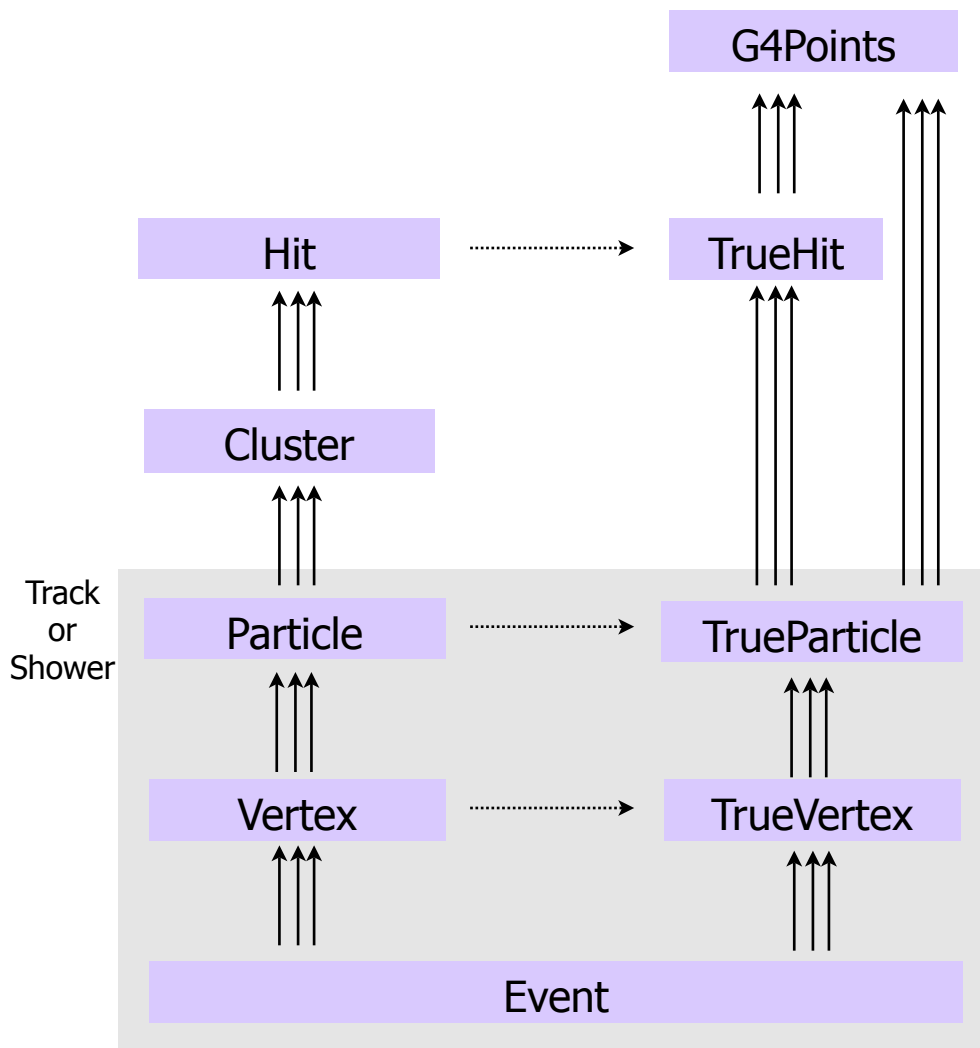
Data Reduction Process

- I'm not familiar with the LArSoft output but I guess it will not differ much from this:



Data Reduction Process

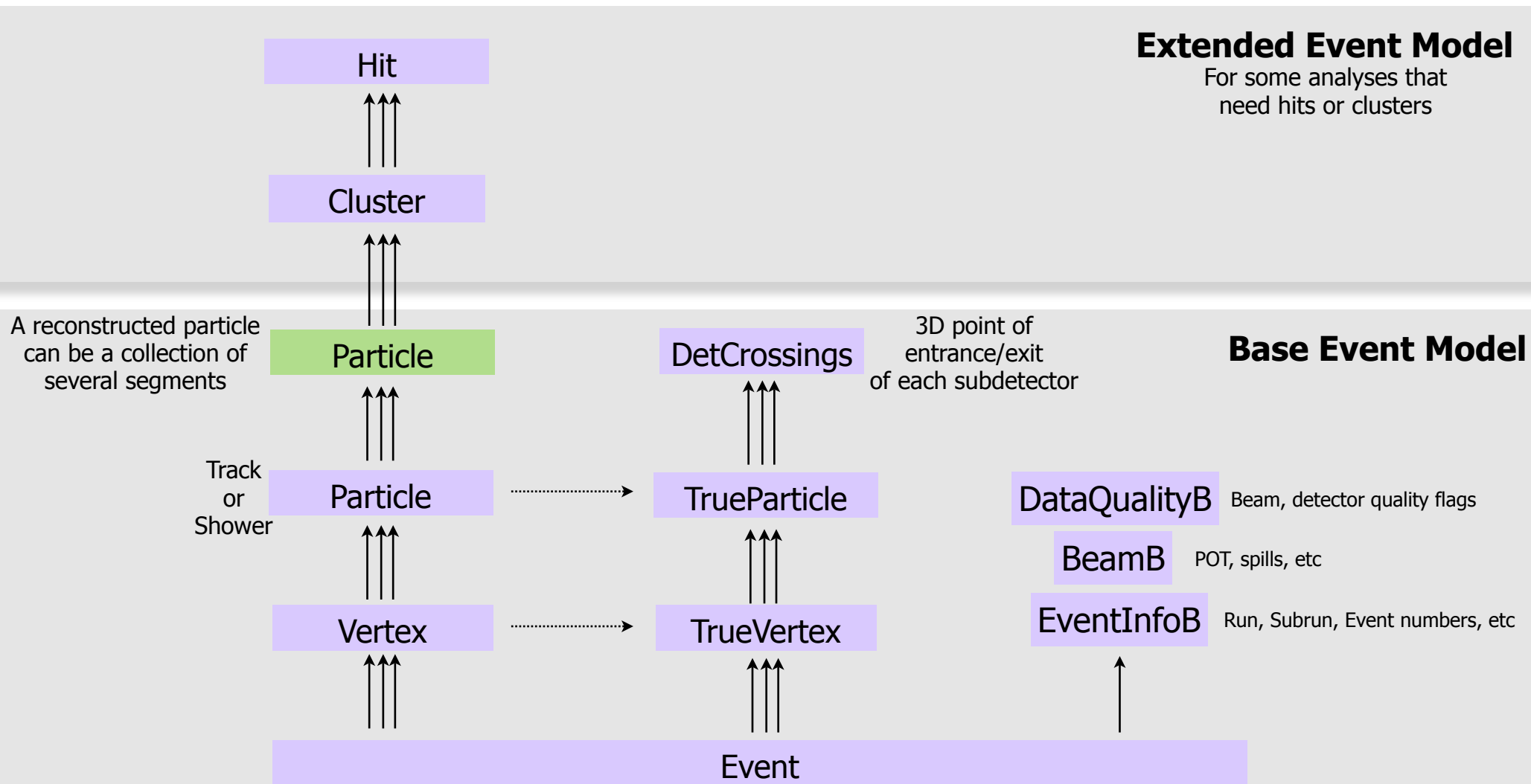
- I'm not familiar with the LArSoft output but I guess it will not differ much from this:



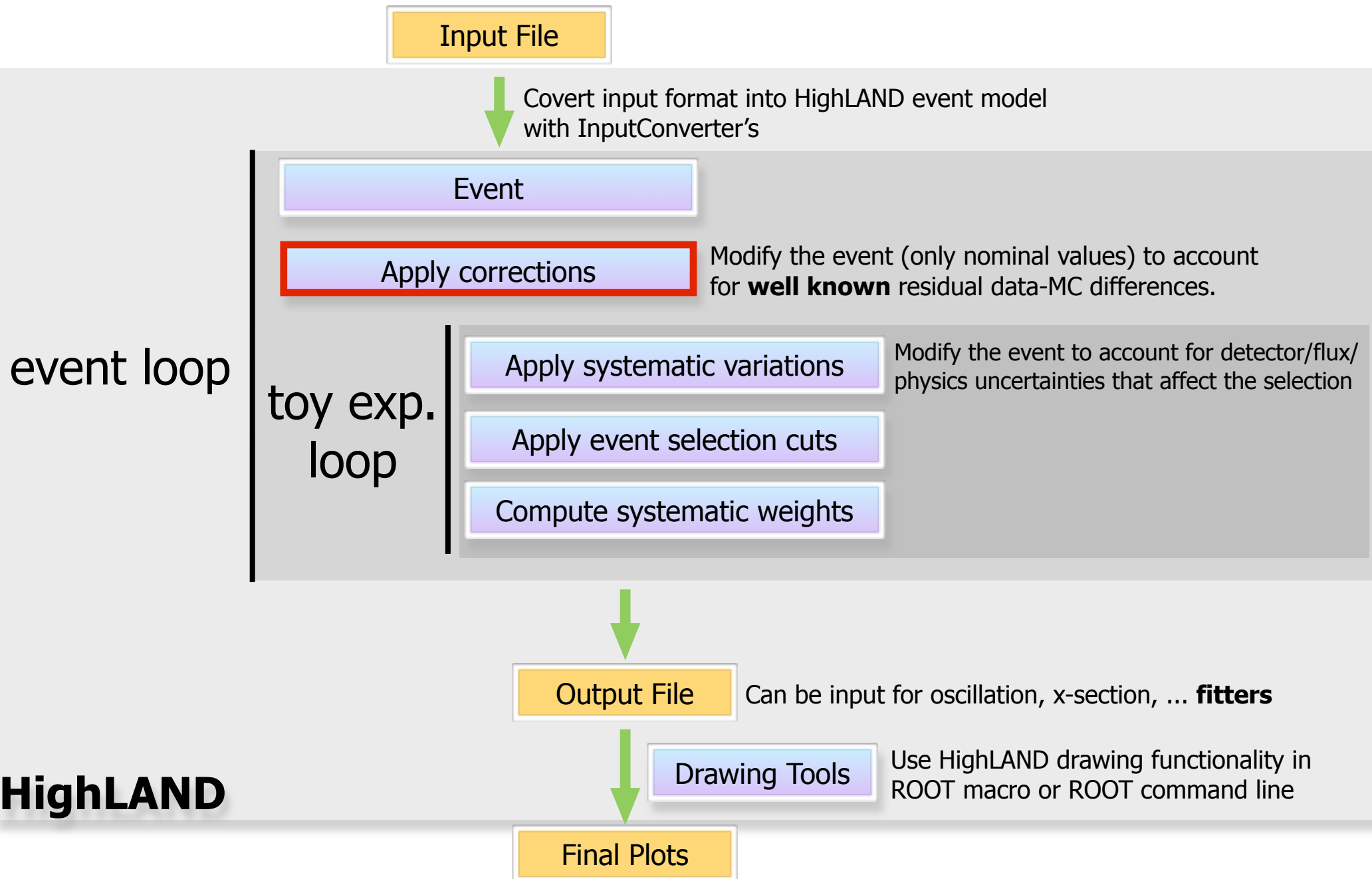
- In general we would need a data reduction process
 - i.e. Don't use hits at analysis level
- In T2K the output of the reconstruction is given to an intermediate package which filters (remove most hits, perform true-reco association, etc) the information and creates a new ROOT file ready for analysis
- This file is the input to HighLAND

T2K-HighLAND event model

In T2K we have something like this
but it could be different for DUNE



Analysis flow



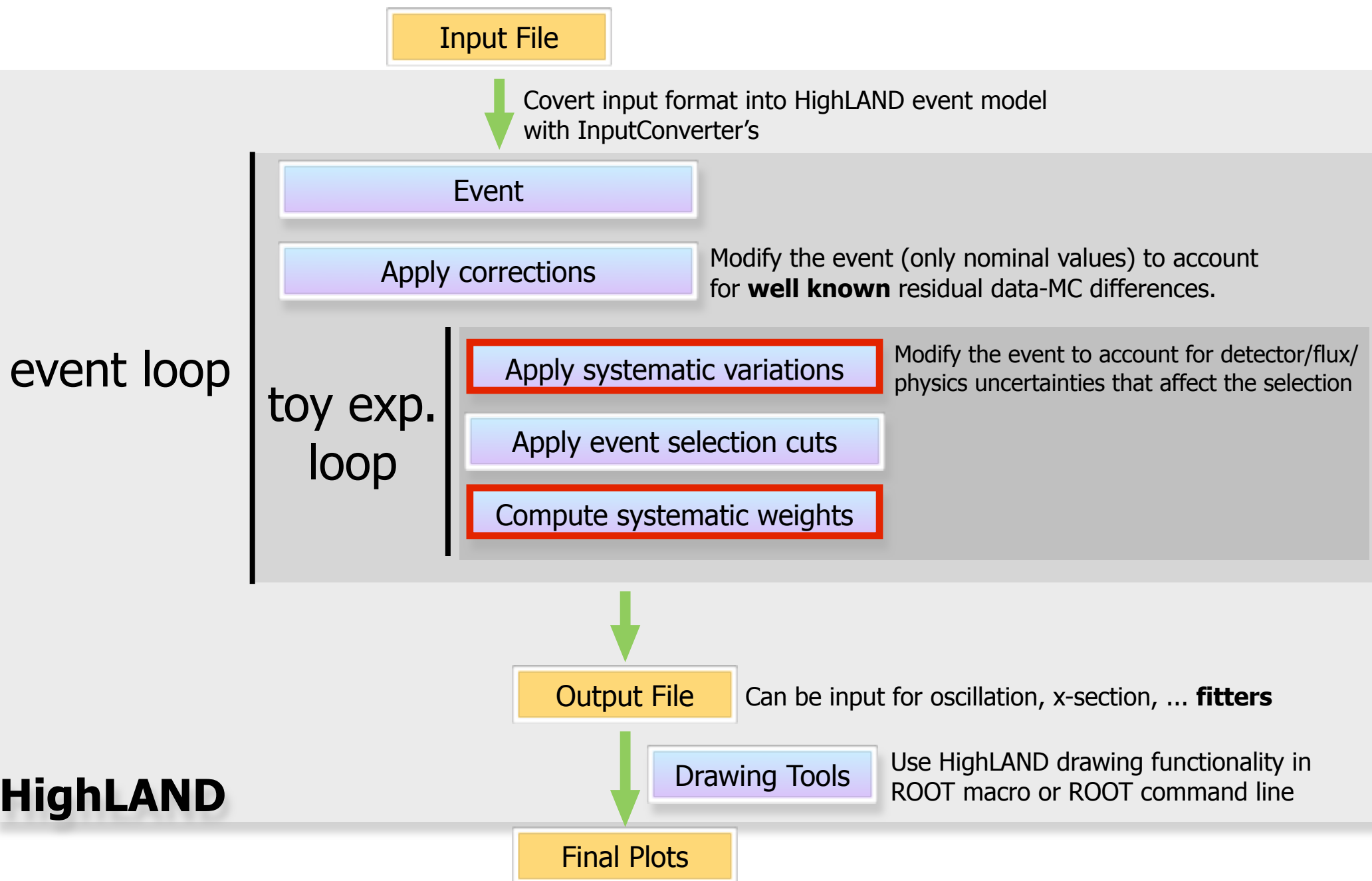
Corrections

- In this step data and/or MC are corrected such that they match each other in detector performance:
- As an example let's imagine that **momentum scale** is different in data and MC
 - Imagine we have a way to quantify this difference
 - We can either propagate this difference as a systematic or correct for it introducing as a systematic only the error on the correction
 - The correction would consist in scaling the momentum of all tracks in the MC to match the momentum scale in data. So we change the **nominal value** of the momentum for each track

Detector/reco optimization

- We can use the **correction** functionality to tweak the output of the reconstruction and perform analysis (selection+systematics) without rerunning the reconstruction
 - Change point or momentum/energy resolution
 - Change momentum/energy scale
 - Change PID information
 - ...
- In that way we can **optimize the detector or reconstruction parameters** without rerunning the reconstruction (at least 3 orders of magnitude slower)

Analysis flow



Systematics

- Systematics are propagated numerically using toy-experiments (pseudo-experiments or virtual analyses)
- Each toy-experiment is defined by a set of random throws (one for each systematic parameter)
- The covariance of the number of events selected in a given bin is computed in the usual way:

$$C_{ij} = \frac{1}{N_{toys}} \sum_{t=1}^{N_{toys}} (N_i^t - \bar{N}_i)(N_j^t - \bar{N}_j)$$

events in bin i for toy t

$$N_i^t = \sum_{e=1}^{N_{events}} W_{e,i}^t$$

average over toys

$$\bar{N}_i = \frac{1}{N_{toys}} \sum_{t=1}^{N_{toys}} N_i^t$$

Two types of systematics

- **Variations:** The event is modified taken into account the set of systematic parameters for a particular toy experiment. Then the entire analysis proceeds on the modified event. For example:
 - Momentum scale (smear the momentum of all tracks in MC around the nominal, see slide 16)
- **Weights:** a weight (which is one by default) is assigned to each event. This weight is computed using event truth/reco info and the systematic parameters for the current toy. This is done in two cases:
 - when the variation method is not possible
 - Imagine for example the track finding efficiency in one of the TPCs. If the efficiency is larger in data than in MC we can't easily add a new track into the MC
 - for global normalization parameters (flux, target mass, etc.)
- In HighLAND they inherit from base class **SystematicBase**

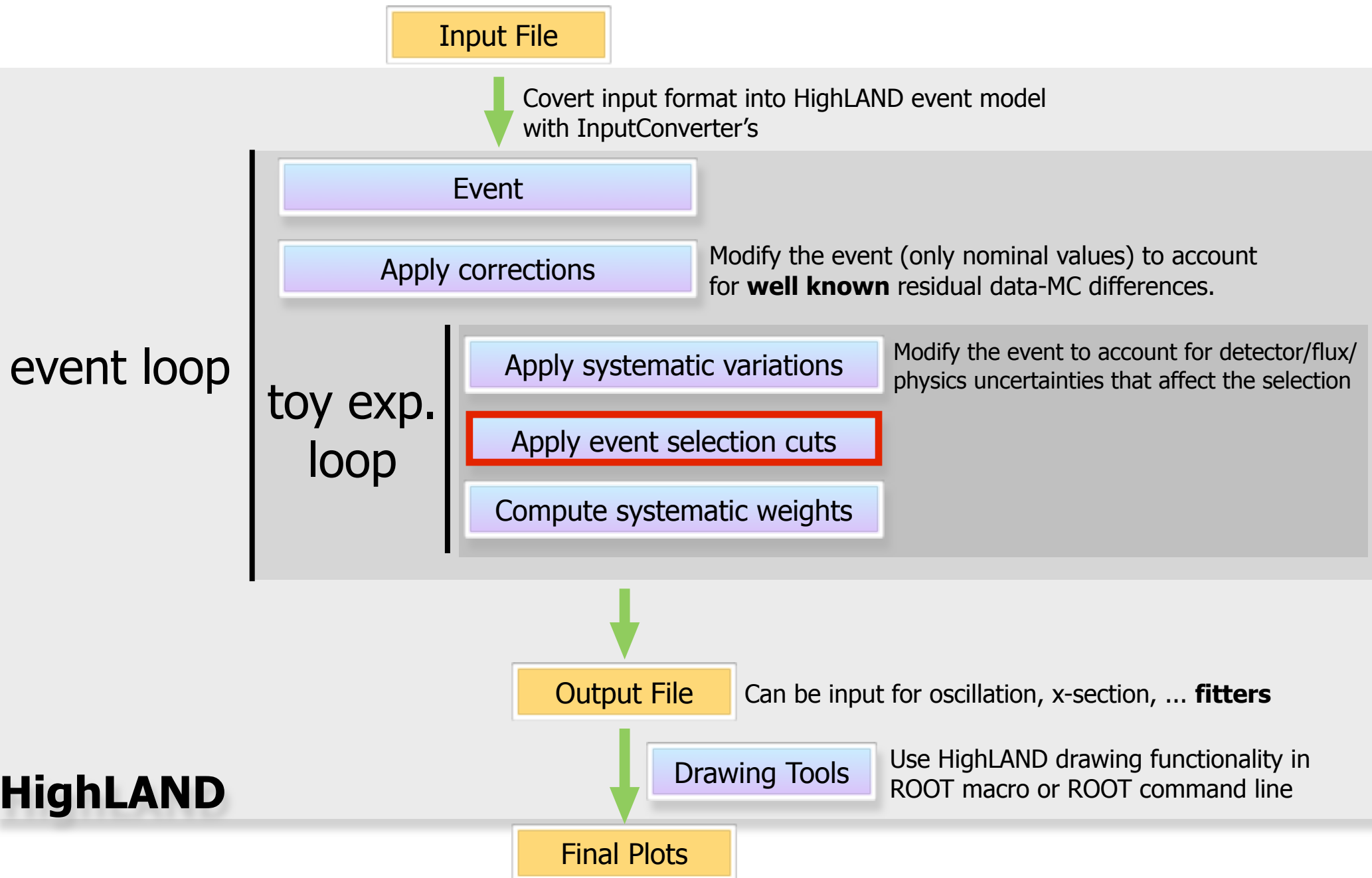
List of systematics in T2K

- This is the list of 31 detector systematics implemented in HighLAND for T2K
- Not all selections use the same systematics but most of them are common to many selections

BFieldDistortionSystematics.hxx
ChargeIDEffSystematics.hxx
ECaleMEnergyResolSystematics.hxx
ECaleMEnergyScaleSystematics.hxx
ECaleMEnergySystematicsBase.hxx
ECalPIDSystematics.hxx
ECalTrackEffSystematics.hxx
FGDECalMatchEffSystematics.hxx
FGDECalSMRDMatchEffSystematics.hxx
FGDHybridTrackEffSystematics.hxx
FGDMassSystematics.hxx
FGDPIDSystematics.hxx
FGDTrackEffSystematics.hxx
FluxWeightSystematics.hxx
MichelElectronEffSystematics.hxx
MomRangeResolSystematics.hxx
MomentumResolSystematics.hxx
MomentumScaleSystematics.hxx

00FVSystematics.hxx
PileUpSystematics.hxx
SIPionSystematics.hxx
SIProtonSystematics.hxx
SandMuonsSystematics.hxx
TPCCLusterEffSystematics.hxx
TPCECalMatchEffSystematics.hxx
TPCFGDMatchEffSystematics.hxx
TPCP0DMatchEffSystematics.hxx
TPCPIDSystematics.hxx
TPCTrackEffSystematics.hxx
TPCVariationSystematics.hxx
ToFResolSystematics.hxx

Analysis flow



Event Selection I

- It's a collection of "steps" (cuts and actions)
 - Each step inherits from the base class **StepBase**
 - It has a single method **Apply**, which returns true or false (only relevant for cuts)
- Each selection inherits from **SelectionBase**, which has a main mandatory method **DefineSteps**

```

//*****
void numuCCSelection::DefineSteps(){
//*****

// Cuts must be added in the right order
// last "true" means the step sequence is broken if cut is not passed (default is "false")
AddStep(StepBase::kCut,    "event quality (good beam/detector)", new EventQualityCut(),    true);
AddStep(StepBase::kCut,    "> 0 tracks ", new TotalMultiplicityCut(),    true);
AddStep(StepBase::kAction, "find lepton candidate", new FindCandidateAction());
AddStep(StepBase::kAction, "find vertex", new FindVertexAction());
AddStep(StepBase::kCut,    "track quality + fiducial volume", new TrackQualityFiducialCut(), true);
AddStep(StepBase::kAction, "find veto track", new FindVetoTrackAction());
AddStep(StepBase::kCut,    "external veto", new ExternalVetoCut());
AddStep(StepBase::kCut,    "muon PID", new MuonPIDCut());

// This is a selection with a single branch, but each branch should have an alias
SetBranchAlias(0,"trunk");

```


Event Selection II

- Example of action (fills the box ...)

```

//*****
bool FindCandidateAction::Apply(AnaEventB& event, ToyBoxB& box) const{
//*****

    // Find leading tracks with good quality and only in FGD FV
    cutUtils::FindLeadingTracks(event,box);

    // For this selection the LeptonCandidate track is the HMN (Highest Momentum Negative) track
    box.LeptonCandidate = box.HMNtrack;
    return true;
}

```

- Example of cut (uses the filled box ...)

```

//*****
bool MuonPIDCut::Apply(AnaEventB& event, ToyBoxB& box) const{
//*****

    // LeptonCandidate must exist
    if (!box.LeptonCandidate) return false;

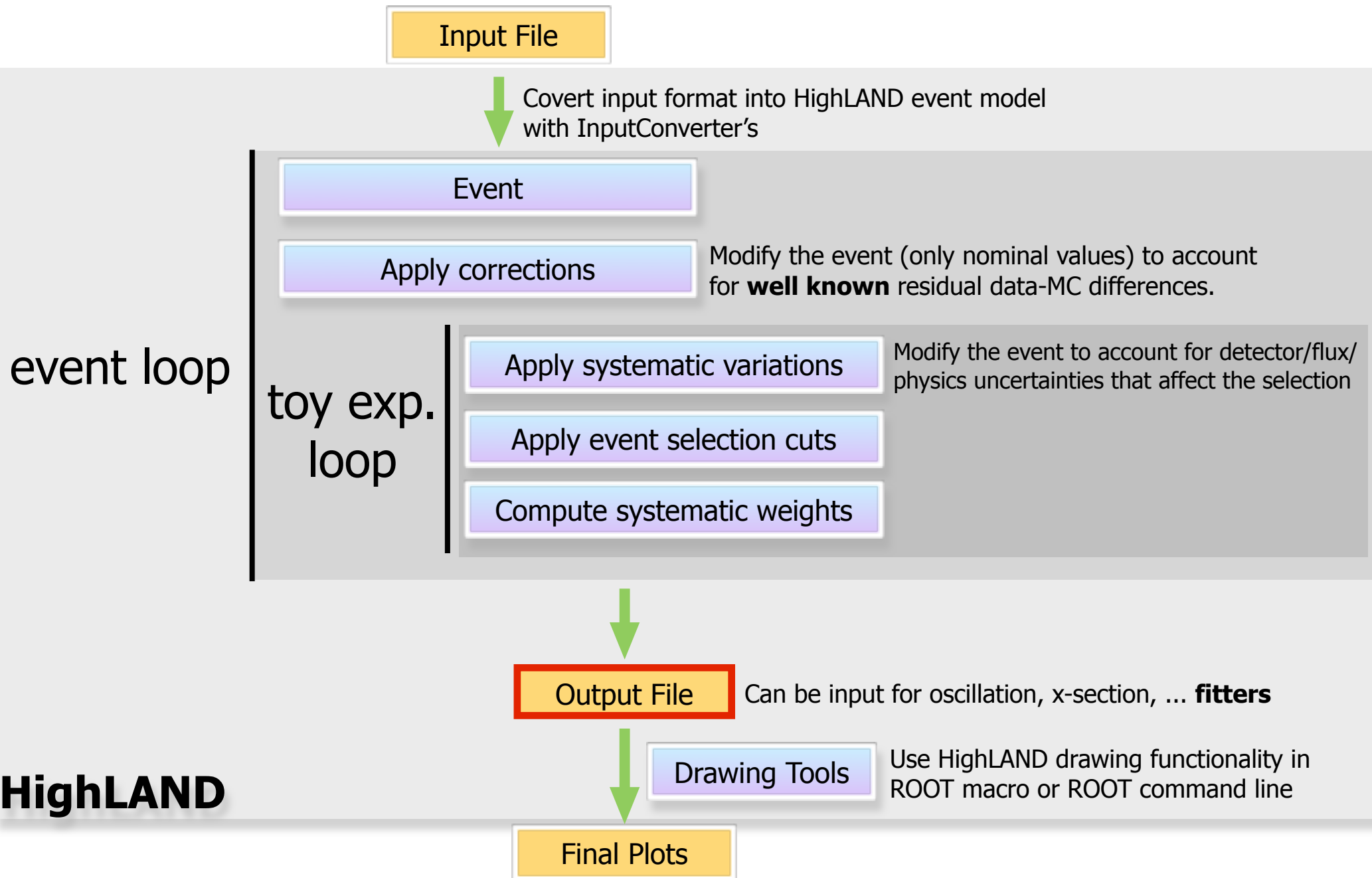
    // And it should have a valid momentum value
    if (box.LeptonCandidate->Momentum < 0.) return false;

    // Apply the PID cut in the utilities namespace to the LeptonCandidate
    return cutUtils::MuonPIDCut( box.LeptonCandidate );
}

```

The box is used
to pass info
from one step
to another

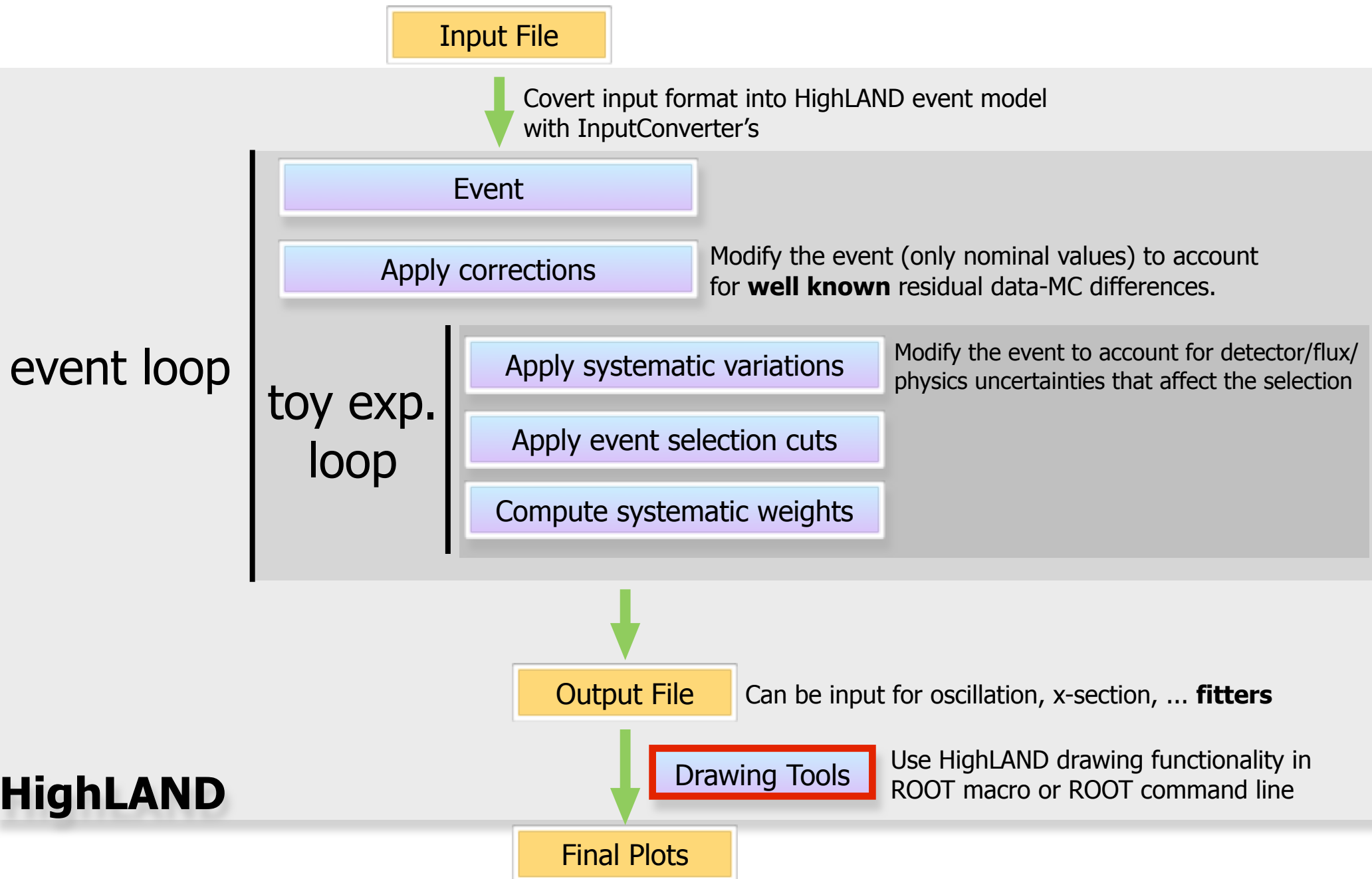
Analysis flow



Output file: The micro-tree

- HighLAND produces an output tree called micro-tree file, which contains several root trees
 - **default**: standard analysis tree containing reco+truth info for all events passing a given cut. This is the nominal selection
 - **syst1, syst2, ..., all_syst** : as the default tree but containing info for each toy experiment for one or several systematics enabled
 - **truth**: tree used to compute efficiencies, containing truth info for all signal events (passing or not the selection)
 - **config**:(Single entry) how the analysis was run
 - Systematics/corrections enabled, input file name, software version, documentation about variables in the analysis tree
 - **header**: (Single entry) POT info

Analysis flow



DrawingTools

- This is one of the framework classes which can be accessed from a ROOT macro or command line
- It is initialized with a micro-tree file (HighLAND output)
- When opening a root session the HighLAND classes are already visible so you just do

```
root [1] DrawingTools draw("microtree.root")
```

- Now you can start doing plots

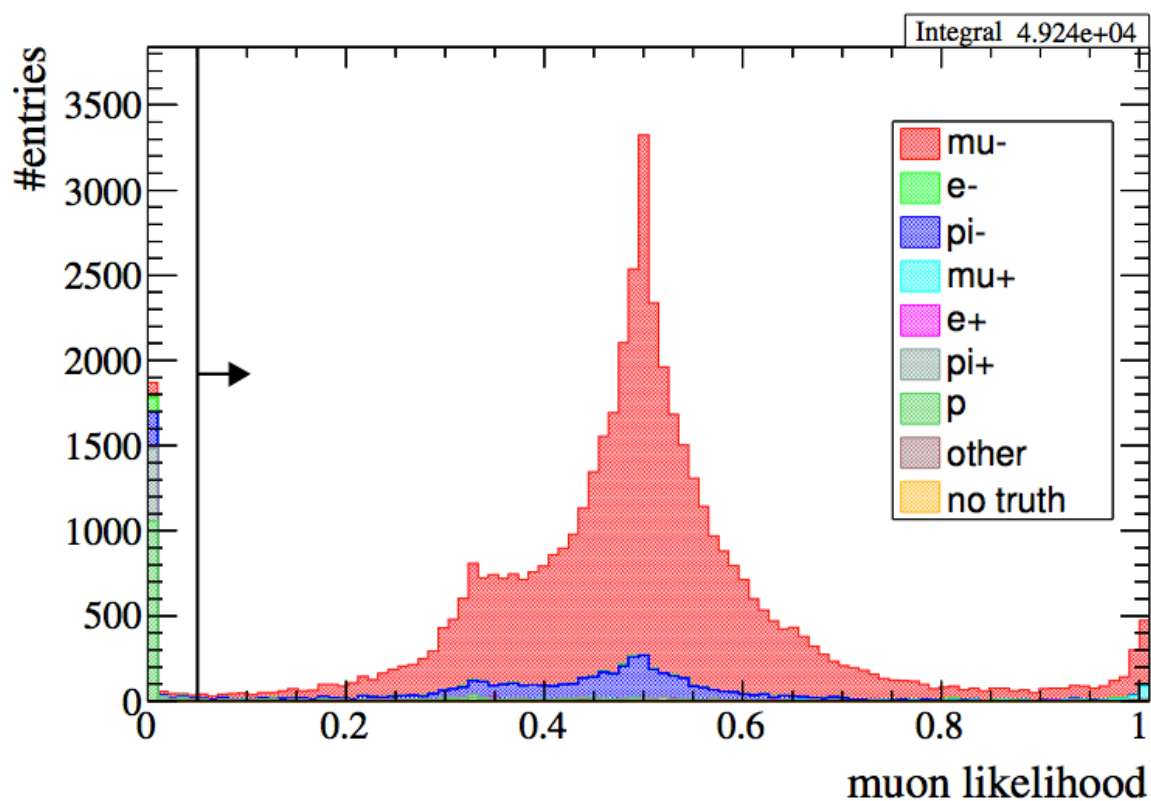
Distributions

- This plot shows the muon PID likelihood before the muon PID cut, broken down in "particle" categories

```

root [2] draw.SetTitleX("muon likelihood")
root [3] draw.SetTitleY("# entries")
root [4] draw.Draw(default,"selmu_likemu",100,0,1,"particle","accum_level>4")
           tree name  variable to plot  binning  color category  events passing cut 4
root [5] draw.DrawCutLineVertical(0.05,true,"r")

```



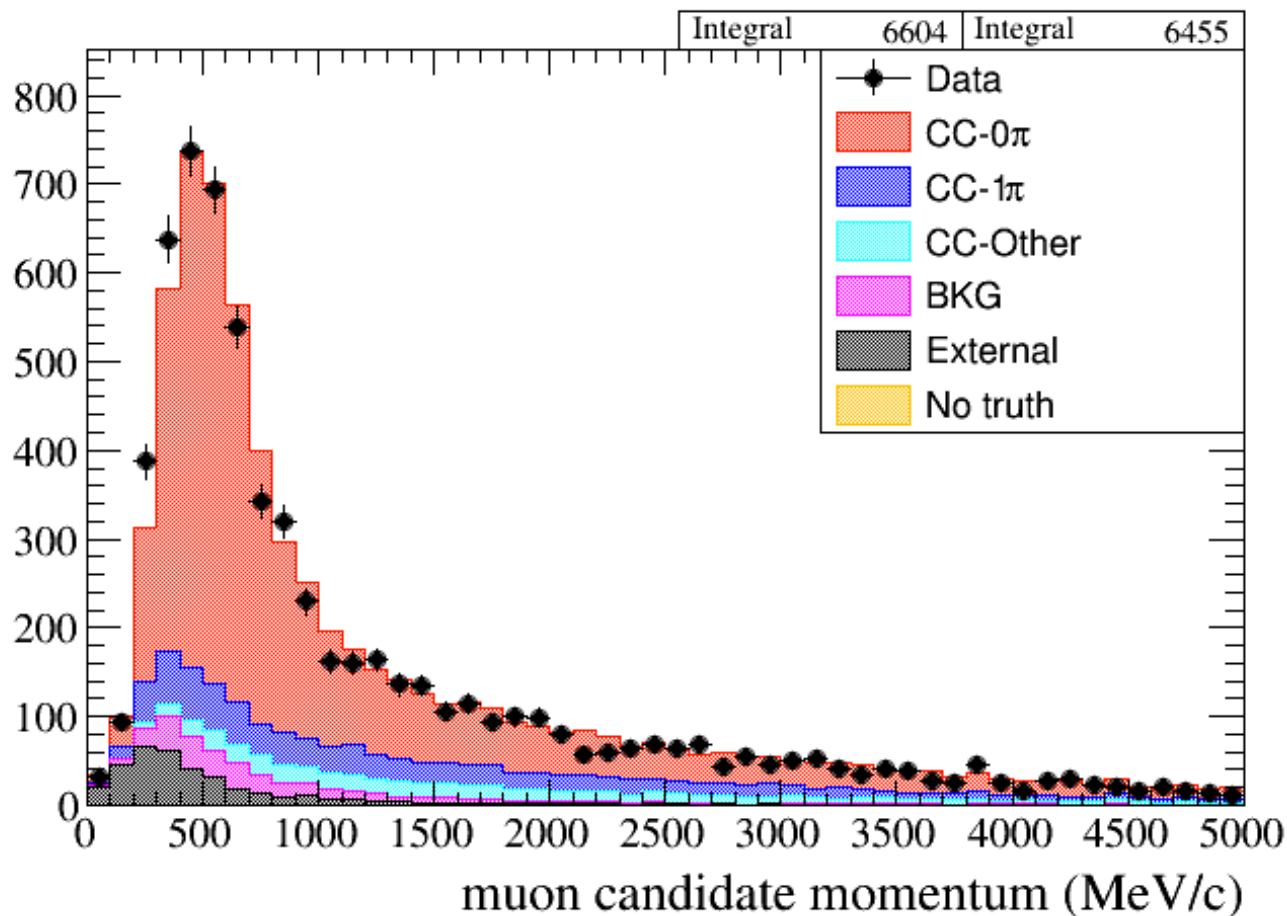
Data/MC comparisons

- We initialize a DataSample class with a micro-tree file

```

root [1] DrawingTools draw("data.root")
root [2] DataSample mc("mc.root")
root [3] DataSample data("data.root")
root [4] draw.SetTitleX("muon candidate momentum (MeV/c)")
root [5] draw.Draw(data,mc,"selmu_mom",50,0,5000,"topology","accum_level>5")

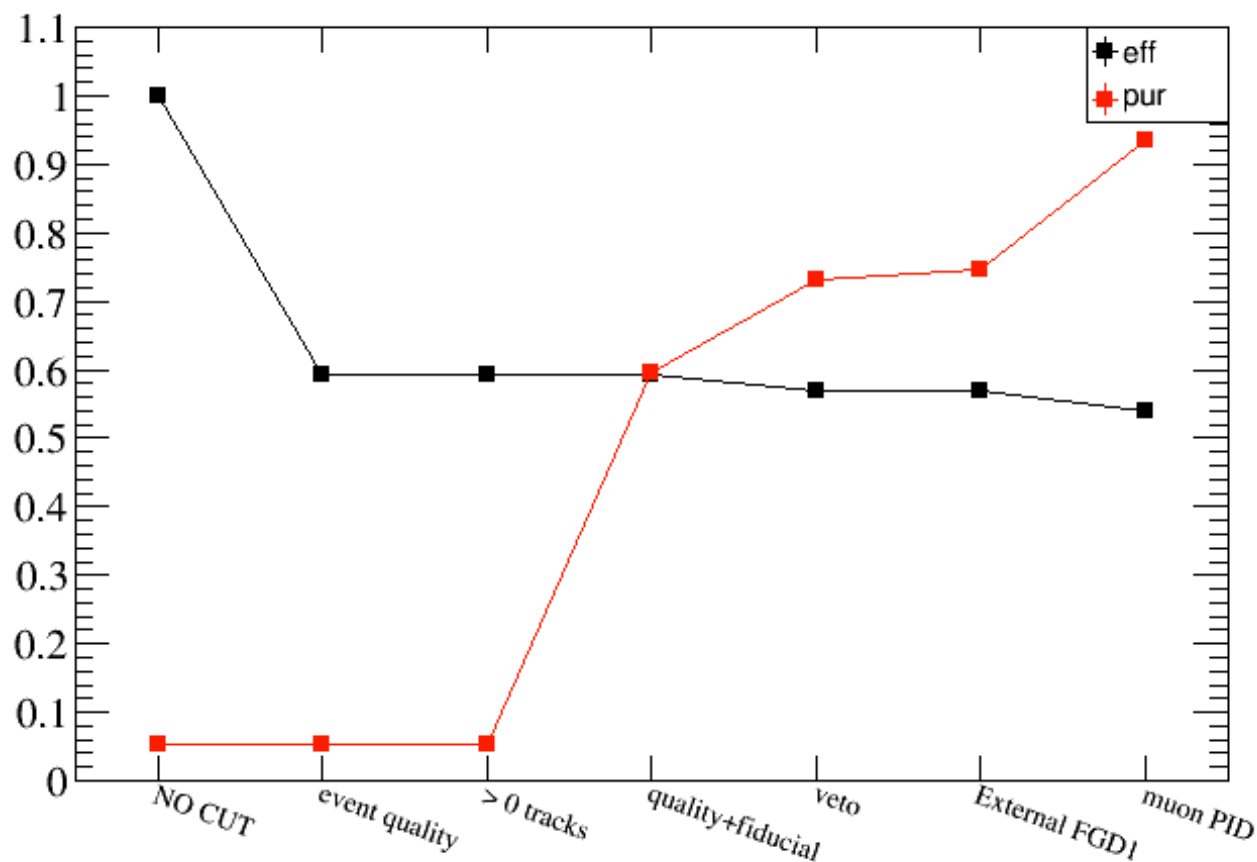
```



Efficiencies & purities

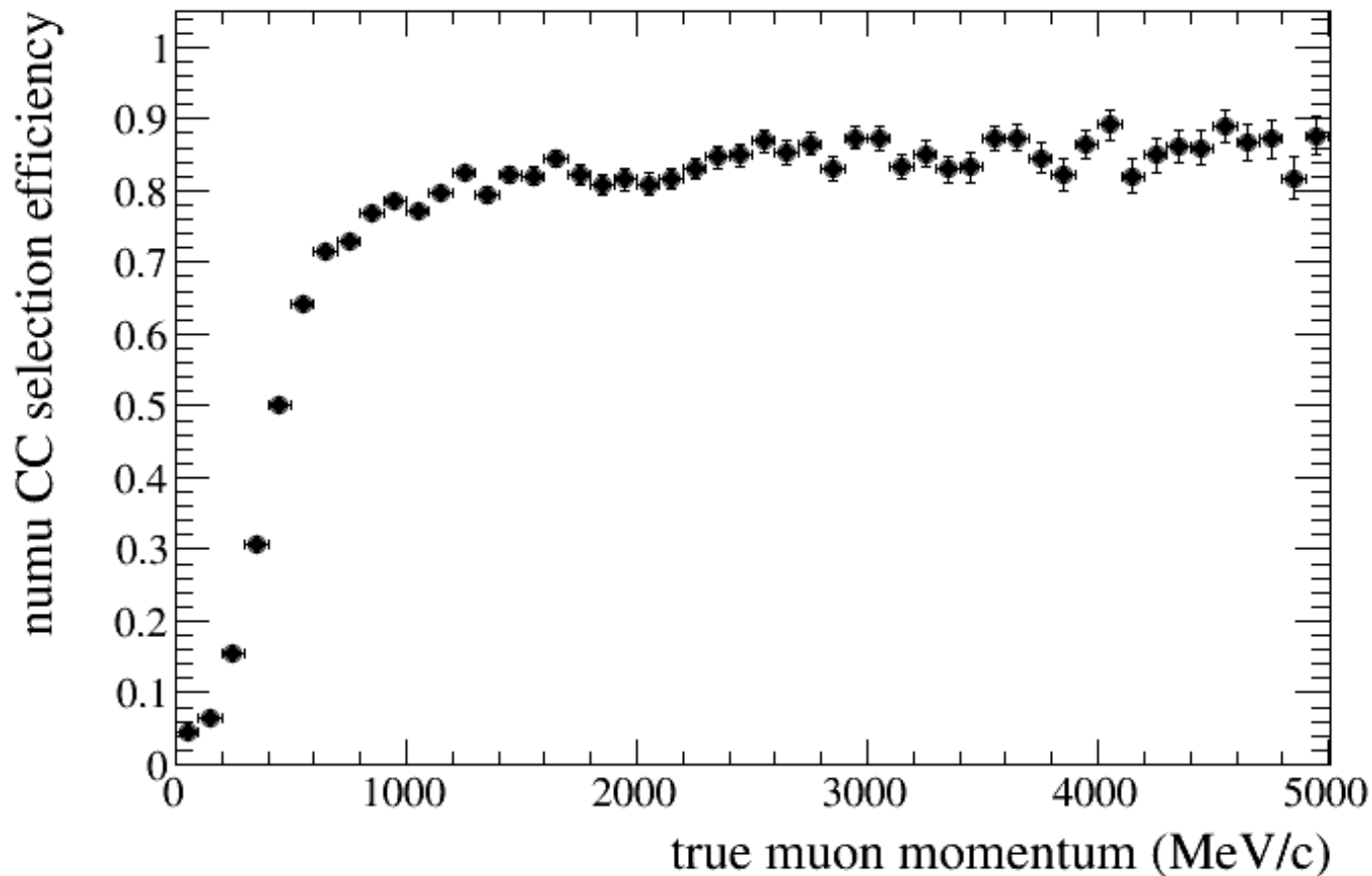
- Efficiency and purity after each cut in the selection

```
root [27] DataSample mc("mc.root")  
root [28] draw.SetTitleY("")  
root [29] draw.DrawEffPurVSCut(mc,"reaction>=0 && reaction<=4")
```



- Efficiency as a function of true muon momentum

```
root [18] draw.SetTitleX("true muon momentum (MeV/c)")
root [19] draw.SetTitleY("numu CC selection efficiency")
root [20] draw.DrawEff(truth,"truemu_truemom",50,0,5000,"accum_level>5","reaction>=0 && reaction<=4")
```



Using Experiment class

```
Experiment exp("t2k");
```

Create Experiment

```
DataSample* data2a = new DataSample("/data3/T2K/DataDir/MicroTrees/numuCCQExsec/data2a_5F.root");
DataSample* data2w = new DataSample("/data3/T2K/DataDir/MicroTrees/numuCCQExsec/data2w_5F.root");
DataSample* data3b = new DataSample("/data3/T2K/DataDir/MicroTrees/numuCCQExsec/data3b_5F.root");
DataSample* data3c = new DataSample("/data3/T2K/DataDir/MicroTrees/numuCCQExsec/data3c_5F.root");
DataSample* data4w = new DataSample("/data3/T2K/DataDir/MicroTrees/numuCCQExsec/data4w_5F.root");
DataSample* data4a = new DataSample("/data3/T2K/DataDir/MicroTrees/numuCCQExsec/data4a_5F.root");
```

Create DataSamples
for data and MC

```
DataSample* mc2a = new DataSample("/data3/T2K/DataDir/MicroTrees/numuCCQExsec/mc2a_5F_all syst.root");
DataSample* mc2w = new DataSample("/data3/T2K/DataDir/MicroTrees/numuCCQExsec/mc2w_5F_all syst.root");
DataSample* mc3a = new DataSample("/data3/T2K/DataDir/MicroTrees/numuCCQExsec/mc3a_5F_all syst.root");
DataSample* mc4a = new DataSample("/data3/T2K/DataDir/MicroTrees/numuCCQExsec/mc4w4a_5F_all syst.root");
DataSample* mc4w = new DataSample("/data3/T2K/DataDir/MicroTrees/numuCCQExsec/mc4w_5F_all syst.root");
```

```
SampleGroup run2a("run2a");
run2a.AddDataSample(data2a);
run2a.AddMCSample("sys", mc2a);
```

```
SampleGroup run2w("run2w");
run2w.AddDataSample(data2w);
run2w.AddMCSample("sys", mc2w);
```

```
SampleGroup run3("run3");
run3.AddDataSample(data3c);
run3.AddMCSample("sys", mc3a);
```

```
SampleGroup run4a("run4a");
run4a.AddDataSample(data4a);
run4a.AddMCSample("sys", mc4a);
```

```
SampleGroup run4w("run4w");
run4w.AddDataSample(data4w);
run4w.AddMCSample("sys", mc4w);
```

Create
SampleGroups
one per period

```
exp.AddSampleGroup("run2a", run2a);
exp.AddSampleGroup("run2w", run2w);
exp.AddSampleGroup("run3", run3);
exp.AddSampleGroup("run4a", run4a);
exp.AddSampleGroup("run4w", run4w);
```

Add SampleGroups
to the Experiment

Final plots with all runs

```

draw.SetTitleX("muon candidate momentum (MeV/c)");
draw.SetTitleY("#entries");
draw.SetAllMCLabel("MC stat error");
draw.SetAllMCStatLabel("MC stat+syst error");
draw.SetMCErrColor(kAzure);
draw.Draw(exp,"selmu_mom",14,pbins,"all","accum_level[0]>7","", "SYS E2");

```

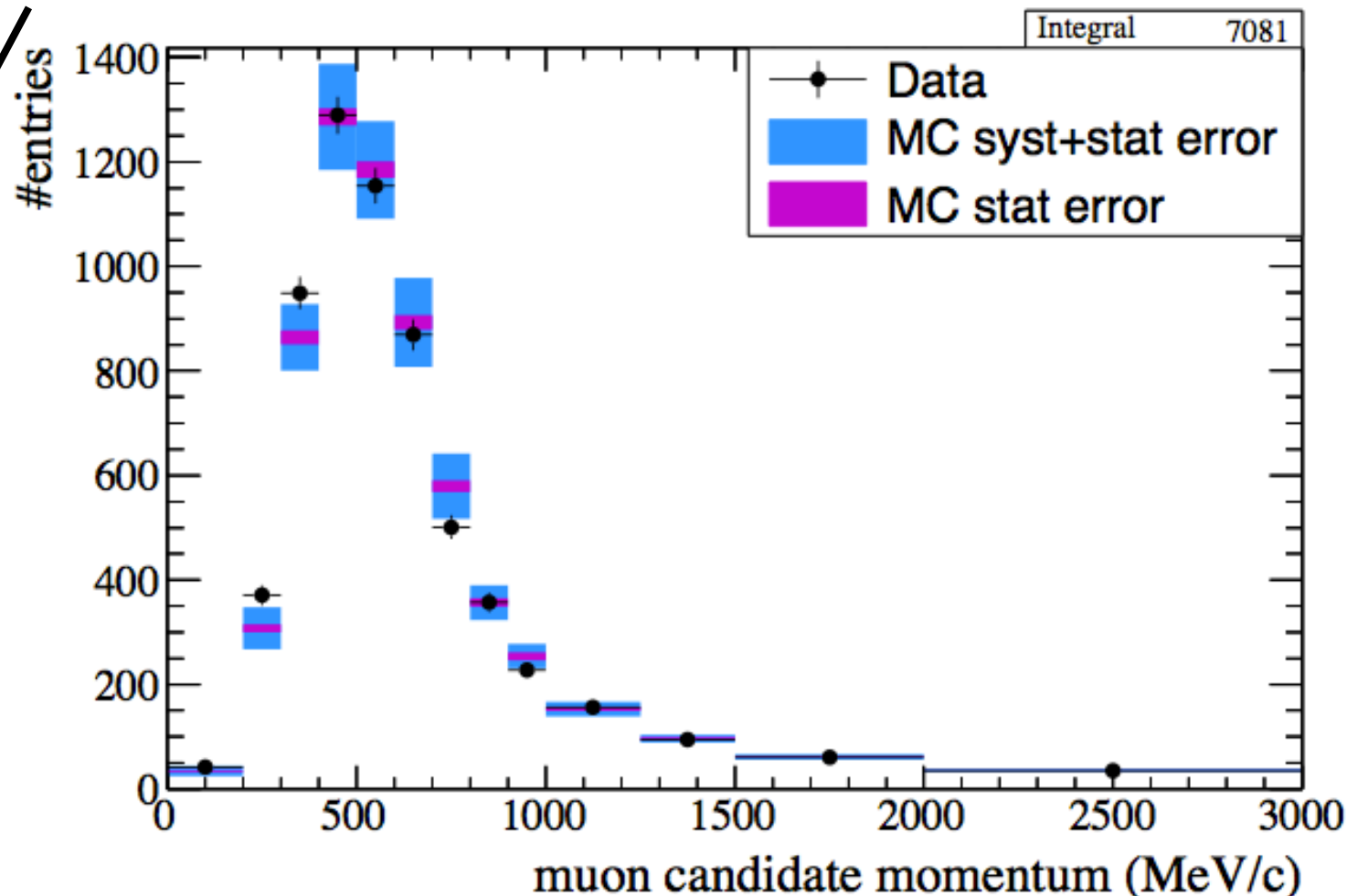
draw systematic

error bars

error style for MC

using
Experiment class

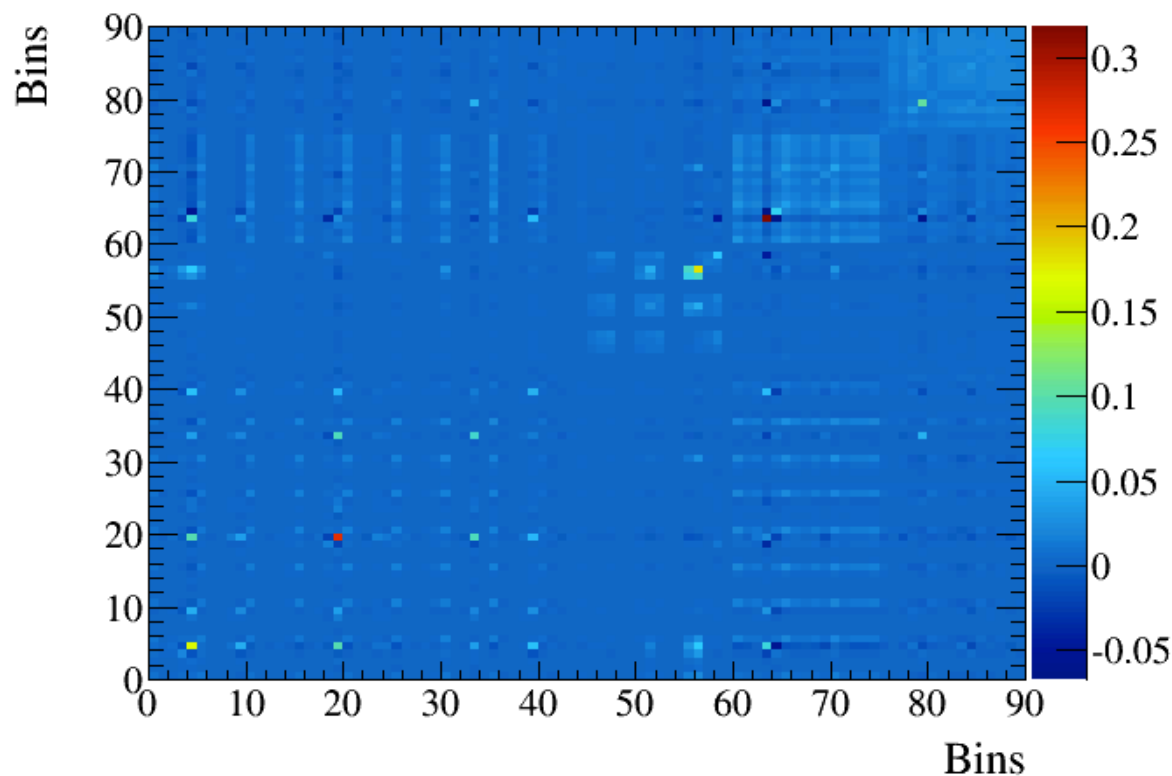
variable
binning



Covariance Matrix

- binning: 3 theta x 5 momentum x 6 samples = 90 bins
- Cov matrix is **computed at plotting time** (all info in the tree). Thus the user can change cuts, binning, etc

1000
throws



official T2K $\nu_{\mu}\text{CC-}0\pi$ x-section

Plans

- We will present a similar talk in several other meetings
 - LBPWG **Oct 24th**
 - Soft/Comp general meeting **Dec 14th**
- If people think that this can be useful for DUNE we will start implementing a prototype. Main ingredients:
 - Analysis Event Model (AEM) for DUNE
 - Need to understand **recon event model** and **analysis requirements**
 - InputConverter for DUNE reconstruction output file
 - Need to understand file format such that it can be converted into the AEM (I'll try to get one of those files)