

DUNE software/computing meeting  
15/12/2015

# HighLAND

## analysis framework

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
- Detailed talks at previous DUNE meetings (backup):
  - FD sim/reco 23/11/2015: <https://indico.fnal.gov/conferenceDisplay.py?confId=10882>
  - LBL 24/11/2015: <https://indico.fnal.gov/conferenceDisplay.py?confId=10861>

# 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

# Why HighLAND and why now?

---

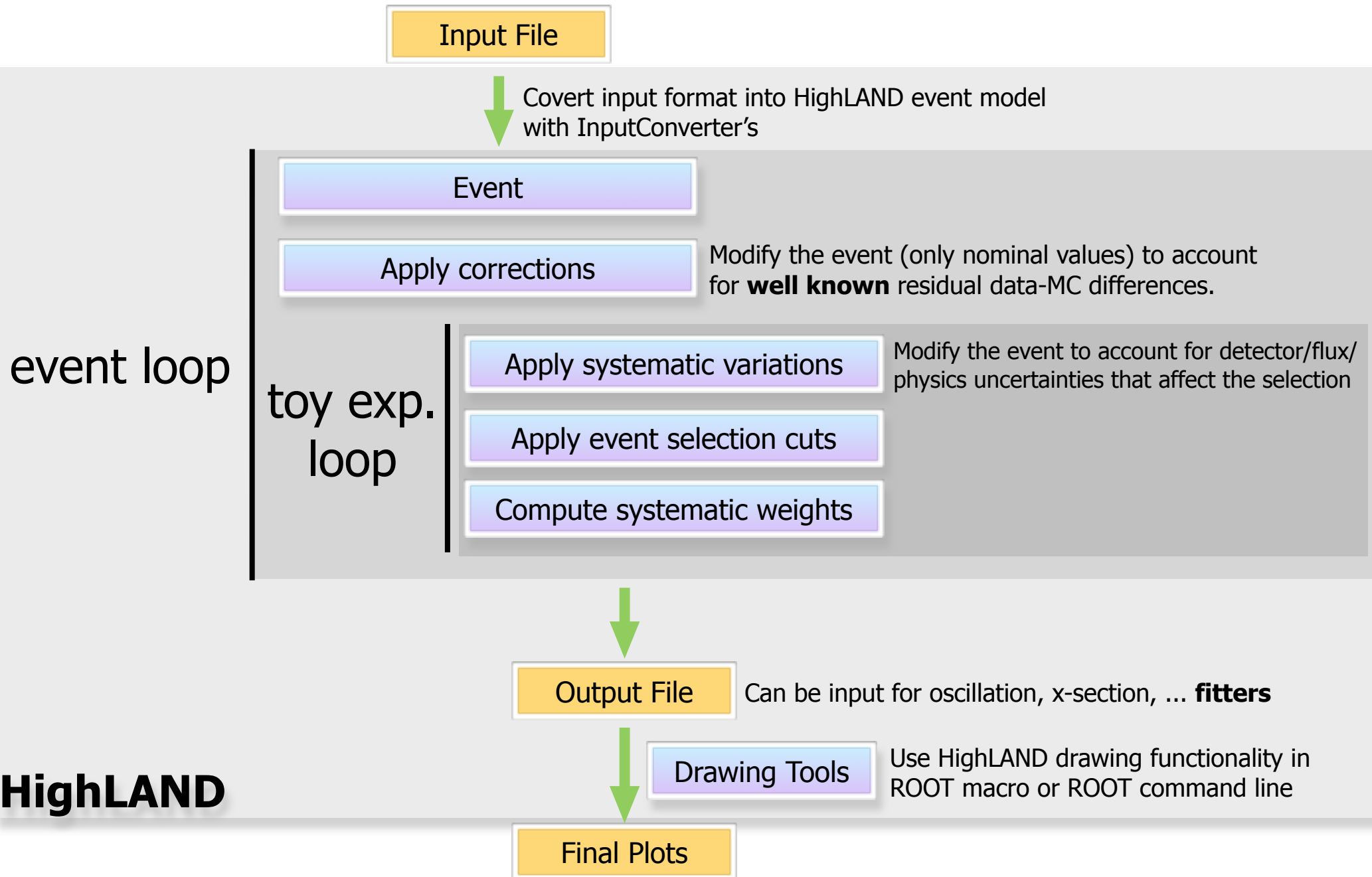
- In T2K we moved into **HighLAND** only 3 years ago, 2 years after the start of the data taking
- Previously there were several frameworks hanging around
  - Information exchange was complicated
  - Comparing analyses almost impossible
  - A nightmare for newcomers
- Now we have many analyses using 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

---

- 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
  - Moving from one group to another should be easier
  - **Correlated 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



# 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**

# Output file

---

- 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, etc
  - **header**: (Single entry) POT info



# Bookkeeping: config tree

```

root [3] draw.DumpFileInfo("test1.root")
===== FILE INFORMATION =====
CMTPATH:    /hep/T2K/nd280rep/ANALYSIS/HIGHLAND2:/hep/T2K/nd280rep/v11r35
HOSTNAME:   unknown
Input file: /hep/T2K/DataDir/oa_nt_beam_90200000-0007_gqbr2v6jhpnw_anal_000_prod6amagnet201011airb-bsdvd01_2.root
----- List of highland2 package Versions -----
#: package          version
0: baseAnalysis    v0r16
1: numuCCAnalysis  v0r16
2: numuCCMultiPiAnalysis v0r12
3: psycheCore      v1r10
4: psycheUtils     v1r10
5: psycheSelections v1r10
6: psycheSystematics v1r10
7: highlandTools   v0r20
8: highlandCorrections v0r12
9: highlandIO      v0r20
-----
nd280 version used to produce original oaAnalysis files: v11r31
----- List of Selections -----
#: name                title                enabled  #branches  force break  index in accum_level
0: kTrackerNumuCCMultiPi  numuCC multi-pionselection  1        3          1            0
-----
----- List of Configurations -----
#: name                NToys  enabled  NSyst  RandomSeed
0: default             1       1        0      -1
1: all_syst            1000    1        15     1
-----
----- List of Corrections -----
#: name                enabled  applied in input
0: pileup_corr         1        0
1: tpcdedx_data_corr   1        0
2: tpcdedx_mc_corr     1        0
3: tpcexpecteddedx_corr 1        0
4: ignorerightecal_corr 1        0
5: dq_corr             1        0
=====

```

The output file contains an extra tree (config) with the analysis configuration

important to know how the analysis was run

# Bookkeeping: config tree

## Cuts in selection

```
root [5] draw.DumpCuts()
-----
      Cuts for selection 'kTrackerNumuCCMultiPi' branch (0) 'CC-0pi'
-----
#  type      title                break  branches
0:  cut      event quality            1      0 1 2
1:  cut      > 0 tracks                  1      0 1 2
2:  cut      quality+fiducial          1      0 1 2
3:  cut      veto                      0      0 1 2
4:  cut      External FGD1           0      0 1 2
5:  cut      muon PID                0      0 1 2
--- split with 3 branches
6:  cut      CC-0pi                  0      0
-----

      Cuts for selection 'kTrackerNumuCCMultiPi' branch (1) 'CC-1pi'
-----
#  type      title                break  branches
0:  cut      event quality            1      0 1 2
1:  cut      > 0 tracks                  1      0 1 2
2:  cut      quality+fiducial          1      0 1 2
3:  cut      veto                      0      0 1 2
4:  cut      External FGD1           0      0 1 2
5:  cut      muon PID                0      0 1 2
--- split with 3 branches
6:  cut      CC-1pi                  0      1
-----

      Cuts for selection 'kTrackerNumuCCMultiPi' branch (2) 'CC-Other'
-----
#  type      title                break  branches
0:  cut      event quality            1      0 1 2
1:  cut      > 0 tracks                  1      0 1 2
2:  cut      quality+fiducial          1      0 1 2
3:  cut      veto                      0      0 1 2
4:  cut      External FGD1           0      0 1 2
5:  cut      muon PID                0      0 1 2
--- split with 3 branches
6:  cut      CC-Other                0      2
-----
```

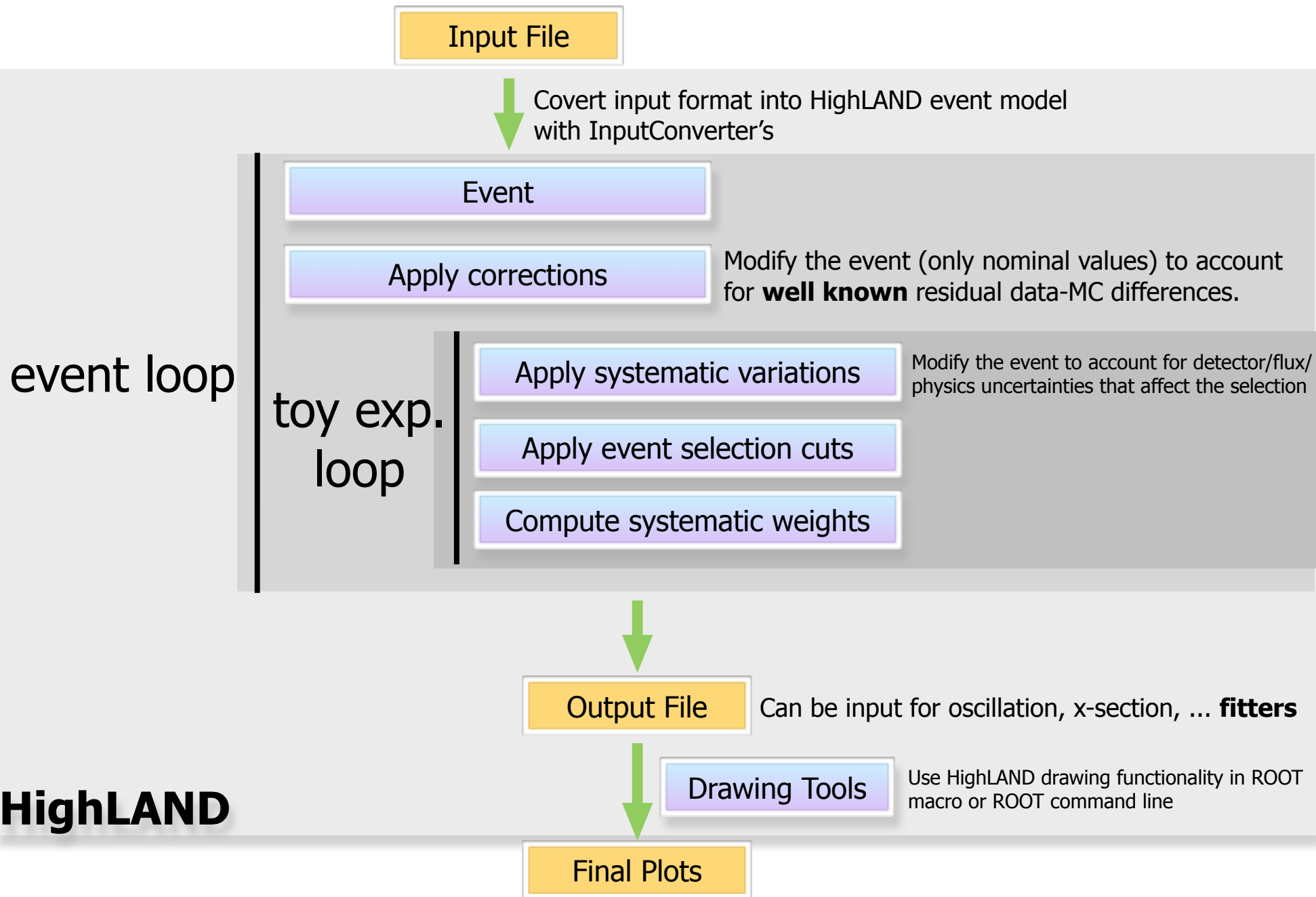
## Enabled systematics

```
root [2] draw.DumpConfiguration("all_syst")
***** Configuration: all_syst *****
enabled:      1
NToys:       1000
NSyst:       15
Random seed: 1

----- List of VariationSystematics -----
#  name                pdf          NPar
0:  kBFieldDist        unknown      1
1:  kMomScale          unknown      1
2:  kMomResol          unknown      10
3:  kTpcPid            unknown      88
4:  kFgdPid            unknown      4

----- List of WeightSystematics -----
#  name                pdf          NPar
0:  kChargeConf        unknown      7
1:  kTpcClusterEff    unknown      4
2:  kTpcTrackEff      unknown      6
3:  kTpcFgdMatchEff   unknown      2
4:  kFgdHybridTrackEff unknown      24
5:  kMichel            unknown      8
6:  kPileUp            unknown      9
7:  kFgdMass           unknown      2
8:  kOOFV              unknown      18
9:  kSIPion            unknown      3
-----
```

# Framework structure



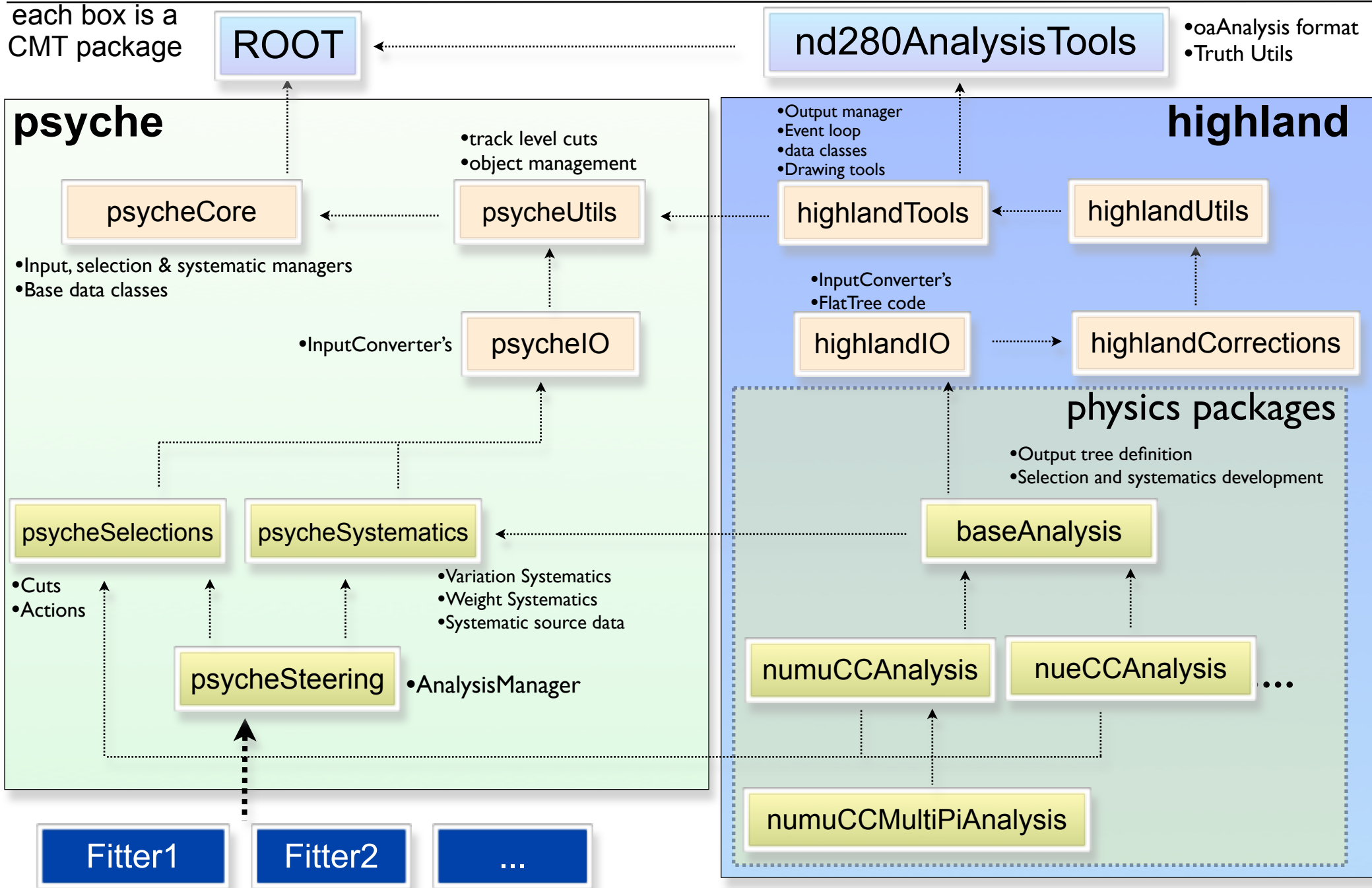
# Framework structure

---

- **HighLAND** framework is divided in two sets of packages:
  - **PSyChE** is the core of HighLAND devoted to event selection and systematic error propagation for
    - External fitters (osc., x-section, ...) and HighLAND
  - **HighLAND**, which extends **PSyChE** with
    - Event loop, extended event model, more input converters, corrections and drawing tools
    - Physics Analysis packages with specific selection/systematics and customized output trees
- HighLANd = High Level Analysis in Dune
- PSyChE = Propagation of Systematics and Characterization of Events

# Package hierarchy in T2K

each box is a  
CMT package



# The code

- There are 157 header files, most of them with an associated source file
- The main framework packages are psycheCore and highlandTools (57 header files)
- Utils and IO packages contain some general code but also detector specific code. We could probably split them

package name	# header files	detector dependent
psycheCore	34	<b>NO</b>
psycheUtils	17	some
psycheIO	2	YES
psycheSelections	17	YES
psycheSystematics	31	YES
psycheSteering	2	some
highlandTools	23	<b>NO</b>
highlandUtils	3	some
highlandCorrections	12	YES
highlandIO	8	some
baseAnalysis	8	YES
<b>TOTAL</b>	157	

# Documentation

- The framework comes with Doxygen documentation

this should be  
complemented by the  
**tutorialAnalysis**  
package

links →

**TZK nd280Highland2 v0r19**

Main Page | Related Pages | Package Maps | Workbook | oaAnalysis format description | Cross referenced source code

nd280Highland2

The highland2 framework

The HighLAND2 framework (HIGH Level Analysis at the Near Detector) is designed to simplify the process of analysing ND280 data. It allows the user to quickly run analyses, plot results, evaluate the impact of systematic errors and much more. HighLAND2 is a second version of the original HighLAND, in which most of the problems of the old framework has been addressed. In particular, the new framework is much faster and can be used by fitters (BANFF, MaCh3) to perform oscillation analyses. To allow this two set of packages have been identified:

- PSyCHE (Propagation of Systematics and CHaracterization of Events) is a high performance set of packages which performs event selection and propagation of systematics.
- HighLAND2: use the psyche packages as core, and it adds in addition Corrections, FlatTree creation, tools for Drawing, etc. In practice PSyCHE is part of the HighLAND2 distribution.

**Analysis flow**

The ND280 analysis flow is in the above diagram. For every event a set of **corrections** are applied to make data and MC agree better. Then a loop over toy experiments is done, each toy experiment will be an independent analysis of the same event, in which the event properties are varied. This allows propagating systematic errors numerically. For each toy experiment, a set of **variation systematics** change some aspect of the data. The **event selection** proceeds then on that modified data. Finally, a **weight** for the event in this particular toy experiment is computed, to account for systematics that cannot be propagated as variations. Those three operations, systematic variations, event selection and systematic weights are performed by PSyCHE. HighLAND2 provides in addition the infrastructure to do the event loop, apply corrections, creating and filling root trees that can be easily analysed using a set of tools for drawing, also provided by HighLAND2.

Each analysis that uses the framework defines its own executable, for example `RunNumuCCRN01ysis.exe` for the `numuCCAnalysis` package. The output of this executable is a root file with several trees. Some of them contain a summary of the variables used in the analysis as well as selection cuts information (those are called micro-trees), while there are also single-entry trees containing more general information such as the amount of POT that was analysed, and configuration information for the analysis. The output file can then be analysed using a simple ROOT macro, and the `DrawingTools` methods provide many useful functions for doing this.

The following related pages contain more detailed documentation:

- An overview of the concepts: data classes, corrections, systematics, configurations, ...
- Structure of the framework and package hierarchy
- Implementing your own analysis package - a tutorial
- How to analyse events with the micro-tree and the DrawingTools
- All you need to know about the DrawingTools
- Common warnings and error messages
- The Flat Tree: how to save time and space
- Propagating and drawing systematic errors
- Handling different oaAnalysis file formats: versioning, different productions, ...
- Committing your analysis package to the repository
- List of Highland2 packages and managers
- From highland to Highland2
- Tips for processing all the ND280 data in highland
- More advanced tips for implementing an analysis

# Repository and releases

---

- The nd280 highland software is managed with **CVS**
  - I guess we will use **Git** in DUNE
- We currently have releases every ~two months
  - vXrYpZ: version revision patch (no patch for standard releases)
  - Which versioning scheme is it used in DUNE ?
- For package management we use **CMT**. A single command is enough to get the entire code

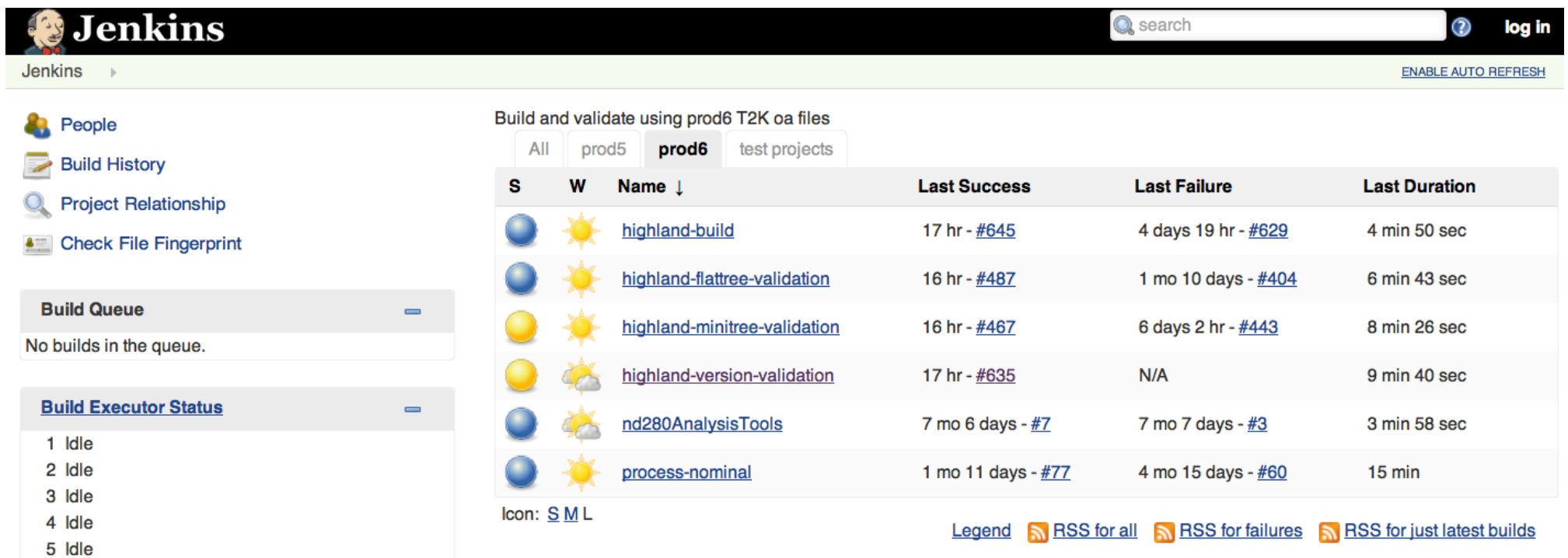
```
cmt co -R nd280Highland2 -r v1r12
```

- What is the DUNE default for package management ?



# Use Jenkins for validation

- We are using Jenkins software for continues integration tests
- Validation tests are triggered by CVS commits
- The framework is continuously validated
  - Releases are much simpler now



The screenshot shows the Jenkins web interface. At the top, there is a search bar and a 'log in' link. Below the search bar, there is a navigation menu with 'Jenkins' and 'ENABLE AUTO REFRESH'. On the left side, there is a sidebar with links for 'People', 'Build History', 'Project Relationship', and 'Check File Fingerprint'. The main content area shows a build queue and a table of recent builds.

**Build Queue**

No builds in the queue.

**Build Executor Status**

1 Idle  
2 Idle  
3 Idle  
4 Idle  
5 Idle

Build and validate using prod6 T2K oa files

All prod5 **prod6** test projects

S	W	Name ↓	Last Success	Last Failure	Last Duration
		<a href="#">highland-build</a>	17 hr - <a href="#">#645</a>	4 days 19 hr - <a href="#">#629</a>	4 min 50 sec
		<a href="#">highland-flatree-validation</a>	16 hr - <a href="#">#487</a>	1 mo 10 days - <a href="#">#404</a>	6 min 43 sec
		<a href="#">highland-minitree-validation</a>	16 hr - <a href="#">#467</a>	6 days 2 hr - <a href="#">#443</a>	8 min 26 sec
		<a href="#">highland-version-validation</a>	17 hr - <a href="#">#635</a>	N/A	9 min 40 sec
		<a href="#">nd280AnalysisTools</a>	7 mo 6 days - <a href="#">#7</a>	7 mo 7 days - <a href="#">#3</a>	3 min 58 sec
		<a href="#">process-nominal</a>	1 mo 11 days - <a href="#">#77</a>	4 mo 15 days - <a href="#">#60</a>	15 min

Icon: [S](#) [M](#) [L](#)

[Legend](#) [RSS for all](#) [RSS for failures](#) [RSS for just latest builds](#)

<https://jenkins-ci.org/>

# Use bugzilla for bug tracking

- nd280 has its own bugzilla. It is a wiki for bug tracking
- I found it essential for highland development
- Are there any plans to use this in DUNE ?

**Bugzilla – Bug List: highland**

Home | New | Search |  Find | Reports | Preferences | Help | Log out anselmo.cervera@cern.ch

Tue Dec 15 2015 04:37:20 PST

**Status:** NEW, ASSIGNED, REOPENED    **Component:** highland2,psyche    **Product:** ND280Software

---

37 bugs found.

ID	Sev	Status	Resolution	Summary
1006	enh	NEW		adding new categories
1014	min	NEW		with >1 branch, DrawEventsVsCut gives wrong values for cuts < MinAccumLevelToSave
1060	min	NEW		Highland2 does not match true information in Nuwro 6b generation
1075	enh	NEW		DisableStep() not implemented in highland2/psyche
1083	min	NEW		DrawingTools messes up some plots in Experiment class
1093	enh	NEW		Using custom EventBox seems too complicated
1122	enh	NEW		not saving RooTrackerVtx in microtrees (ex processing over prod 6 flattrees is 3 times faster than prod 5)
1124	enh	NEW		New FlatTree saving AnaSpill class
1144	enh	REOP		renaming of some common microtree variables
1147	enh	NEW		Automate psyche/highland2 flux tuning
1148	min	NEW		Are mass systematics treated properly?
1152	min	NEW		FGD1 and FGD2 indexing in systematics

<https://www.bugzilla.org/>

# Plans

---

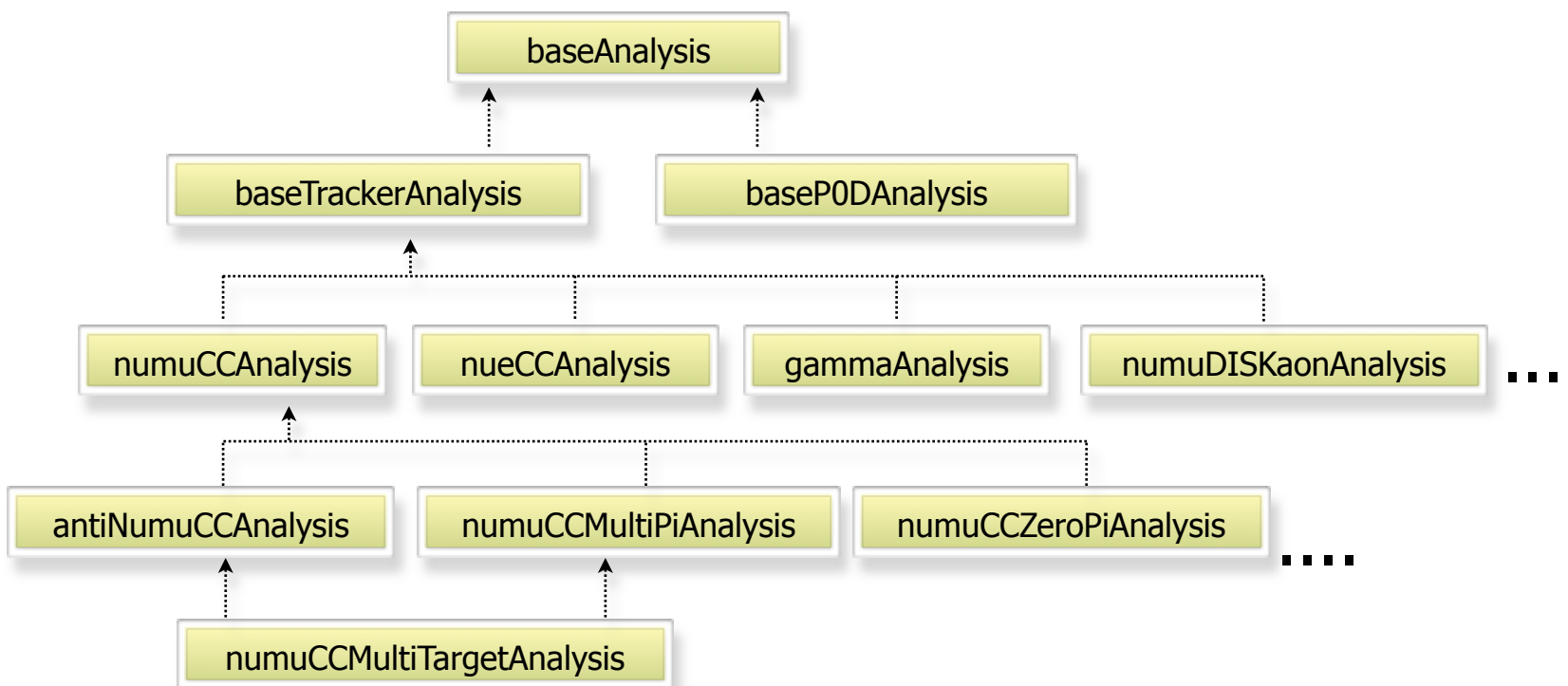
- I'll start implementing a prototype ASAP. Main ingredients:
  - Analysis Event Model (AEM) for DUNE
    - Need to understand **recon event model** and **analysis requirements**
  - InputConverter for DUNE reconstruction output file
    - Need to understand file format such that it can be converted into the AEM
  - The idea is to take the T2K code, remove any T2K specific stuff (shouldn't be much effort) and add DUNE specific stuff (above), if possible fully decoupled from the core framework
- We can discuss it further at the collaboration meeting

**backup**

# A hierarchy of analyses

## Analysis hierarchy in T2K ND

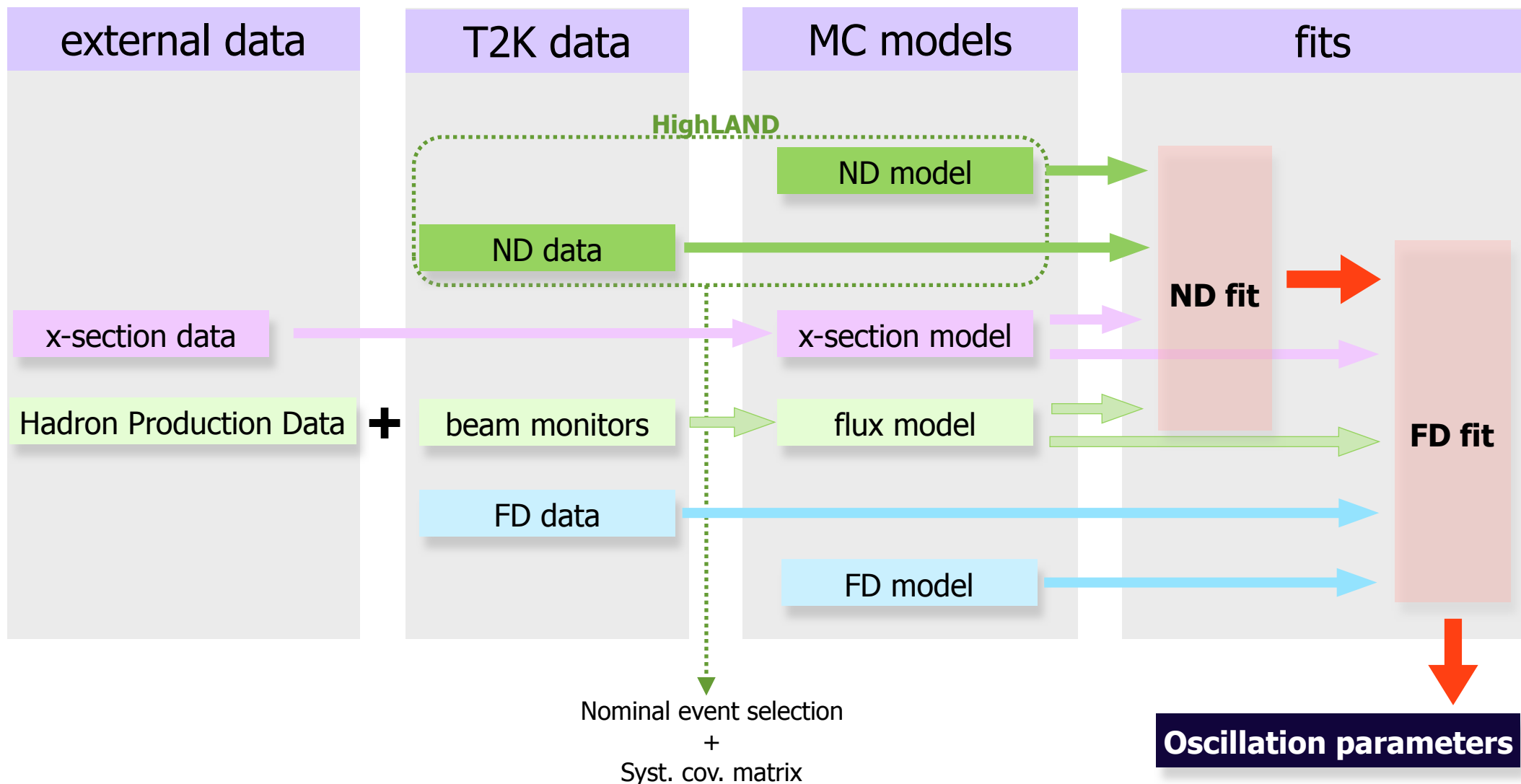
only few  
packages  
shown here  
(>20)



- 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

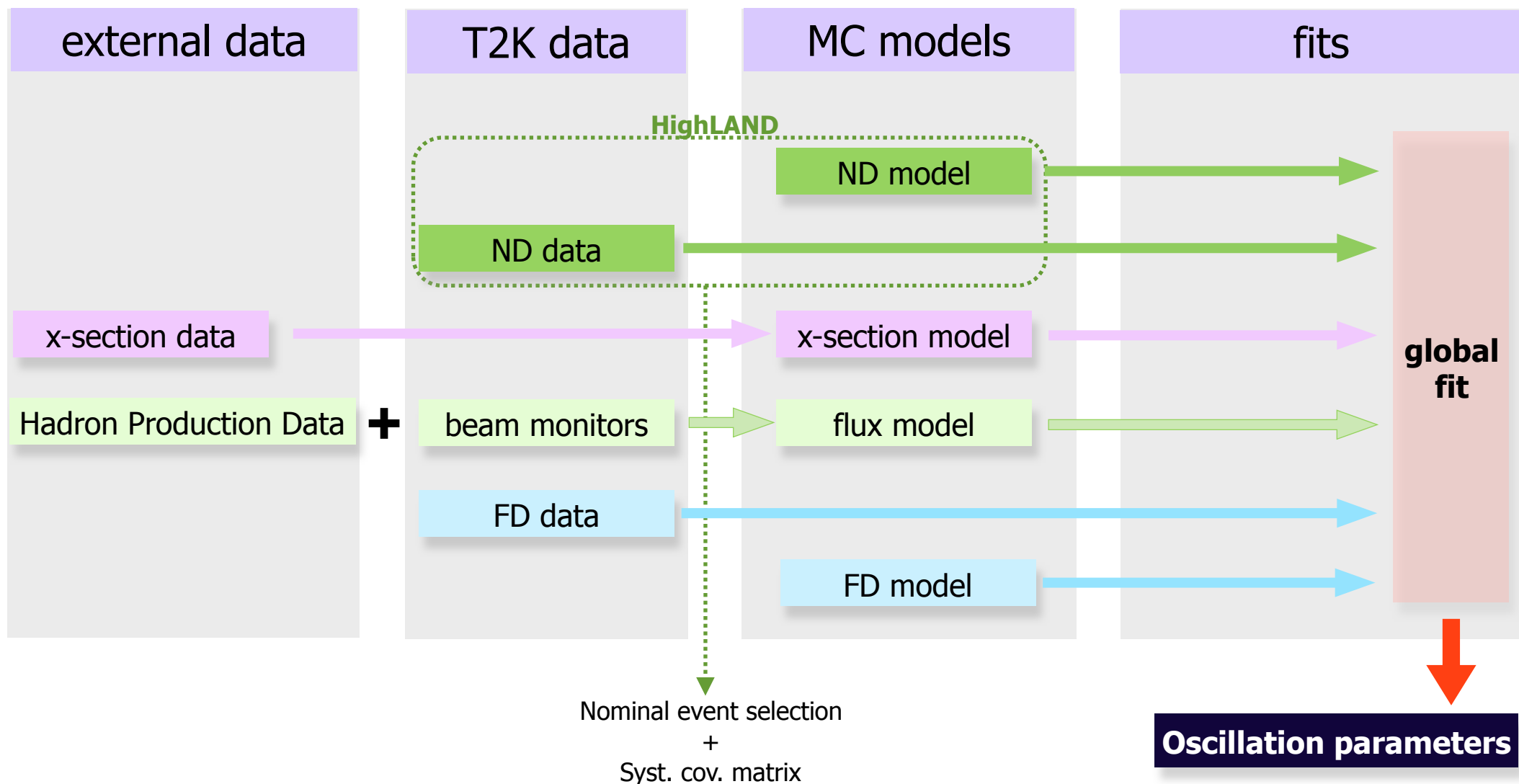
# VaLOR oscillations in T2K

- In T2K HighLAND is only used for ND event selection and systematic error propagation



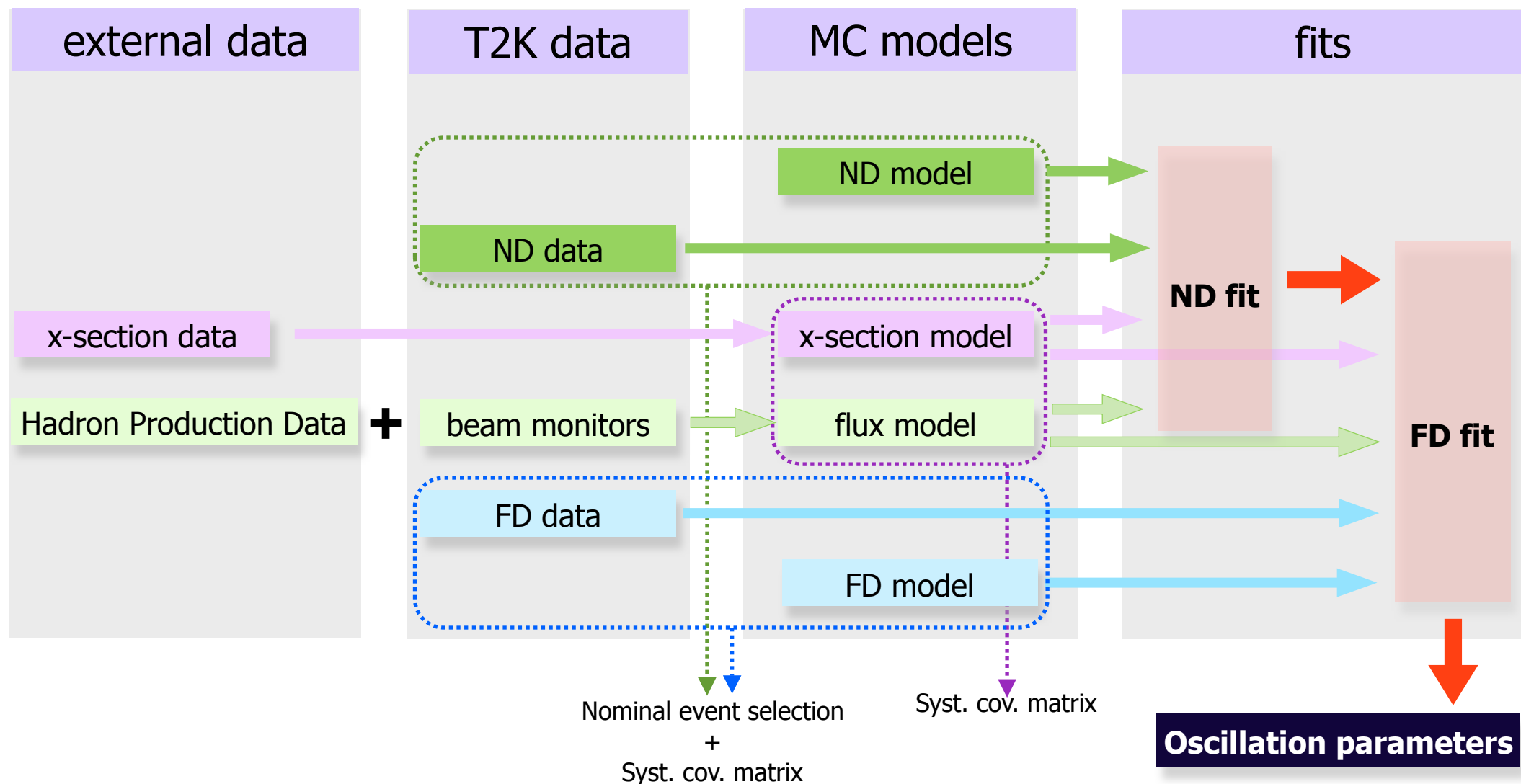
# MaCh3 fit in T2K

- We have another fitter using a Markov Chain MC method using all inputs simultaneously



# In DUNE

- We could also use it for FD selection/systematics + correlations with ND, x-section and flux systematics



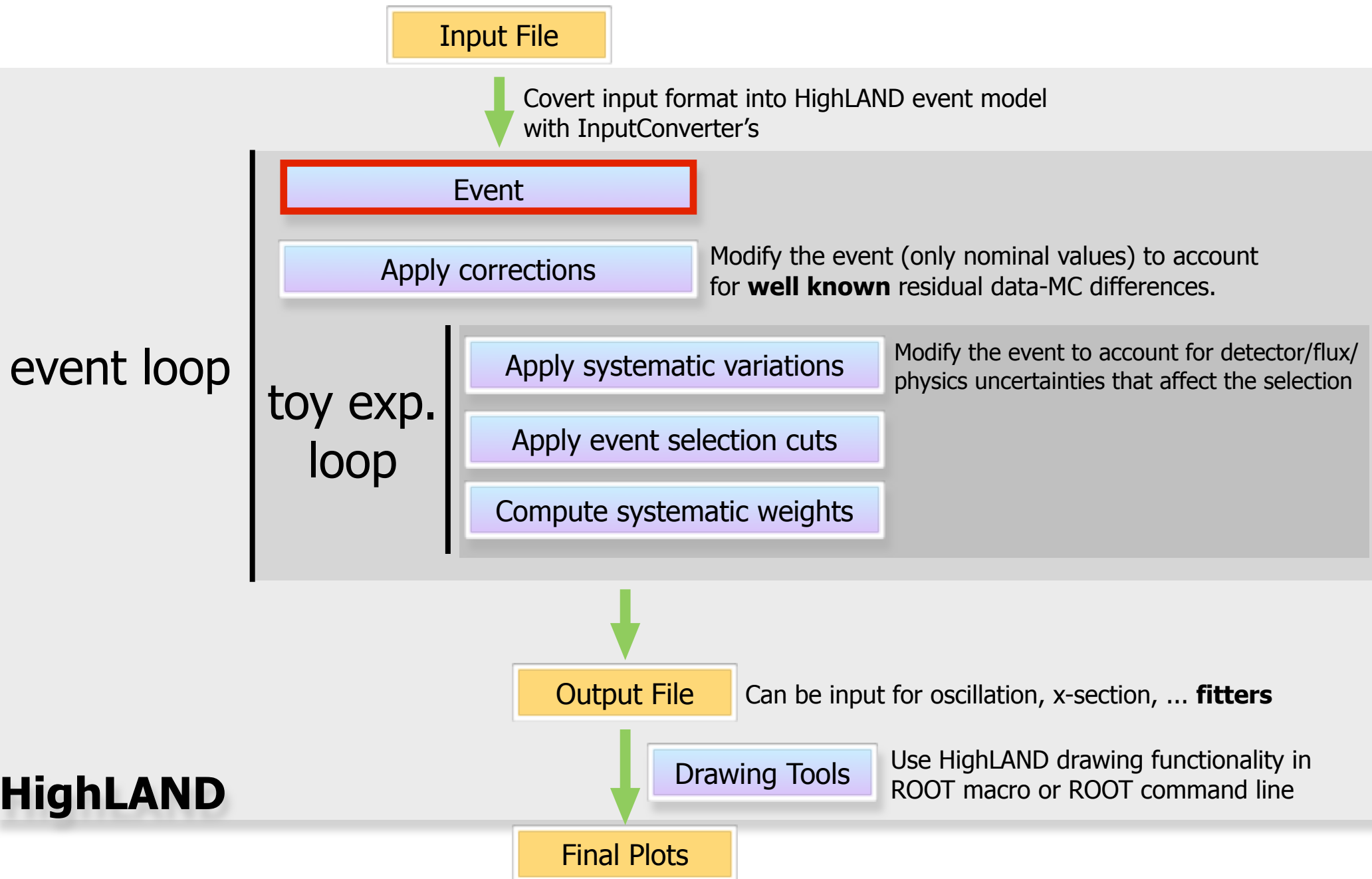


# Two ways of using HighLAND

---

- In T2K we have two ways of propagating systematics with **HighLAND** for oscillation analyses
  1. HighLAND can be used to produce a nominal selection + a covariance matrix, which will be later used by fitters
    - Toy Experiments (random throws) are generated internally by HighLAND
    - A cov. matrix assumes gaussian errors !!!!
  2. Or can be used directly by the fitters:
    - Toy experiments generated by fitters. For each toy HighLAND is called by the fitter to get the results of the selection
    - In T2K we want to move to this

# Analysis flow



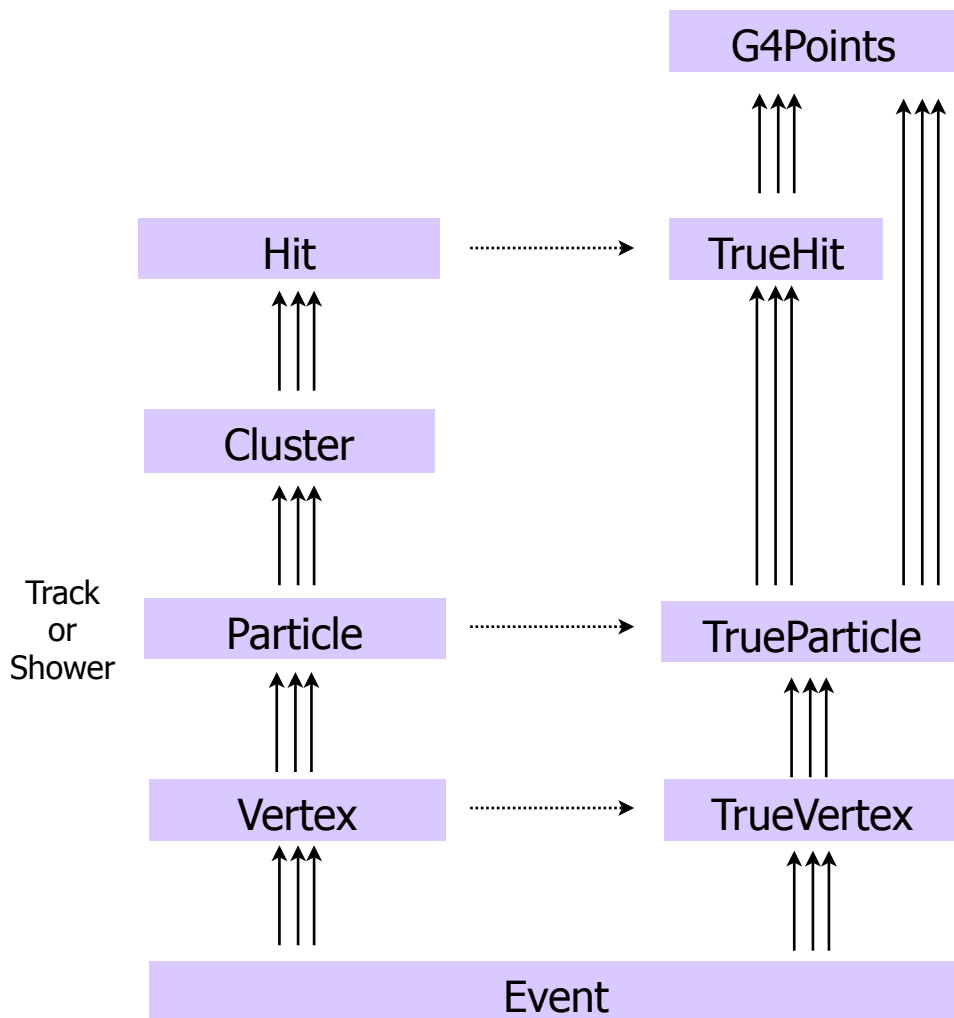
# Event Model

---

- 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)

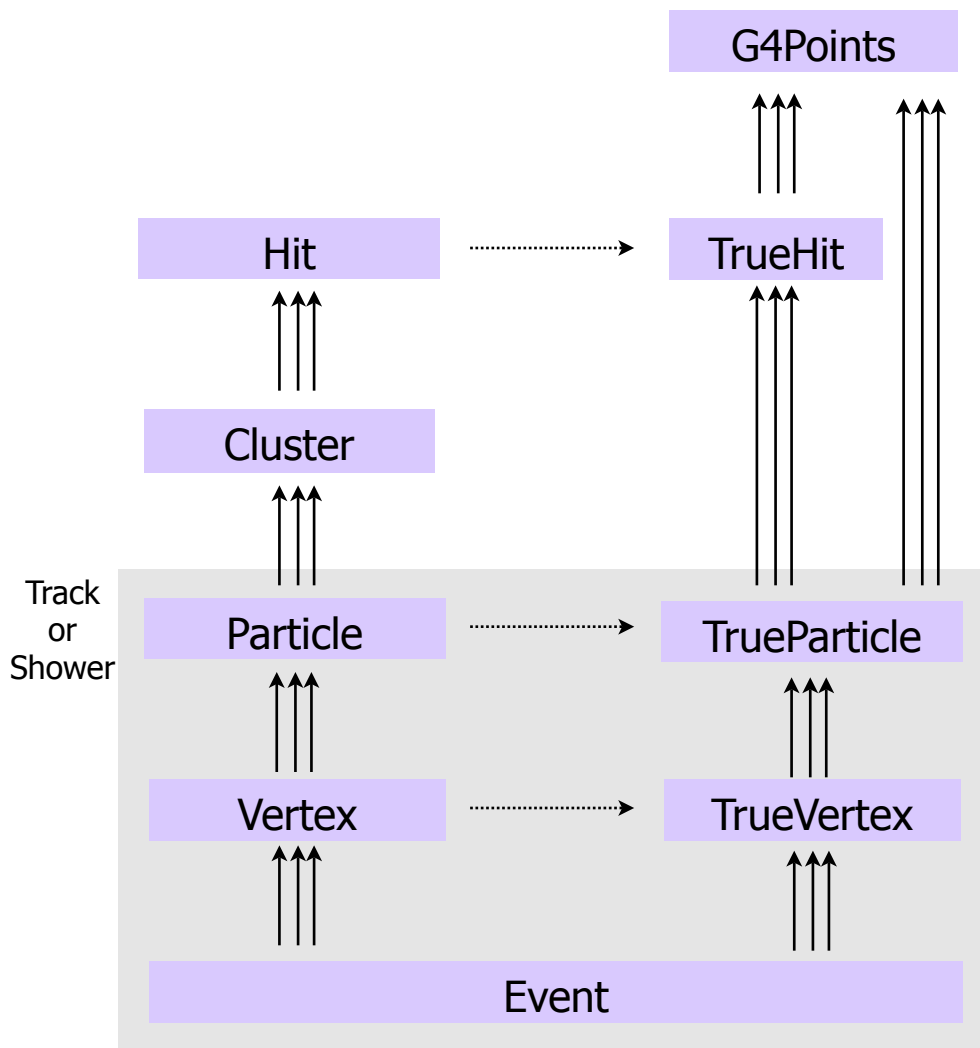
# Data Reduction Process

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



# Data Reduction Process

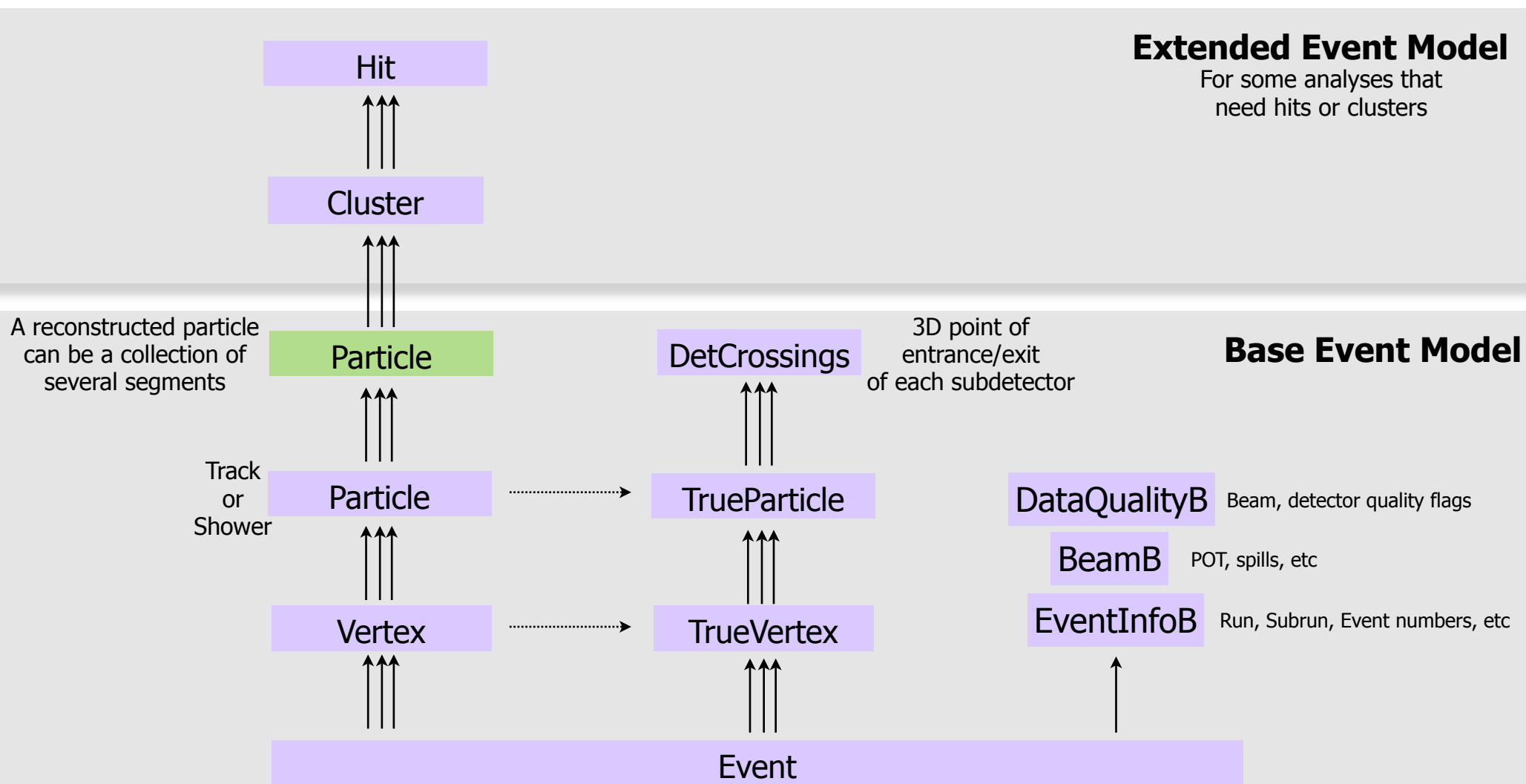
- I'm not familiar with the LArSoft output but I guess it will not differ much from this:



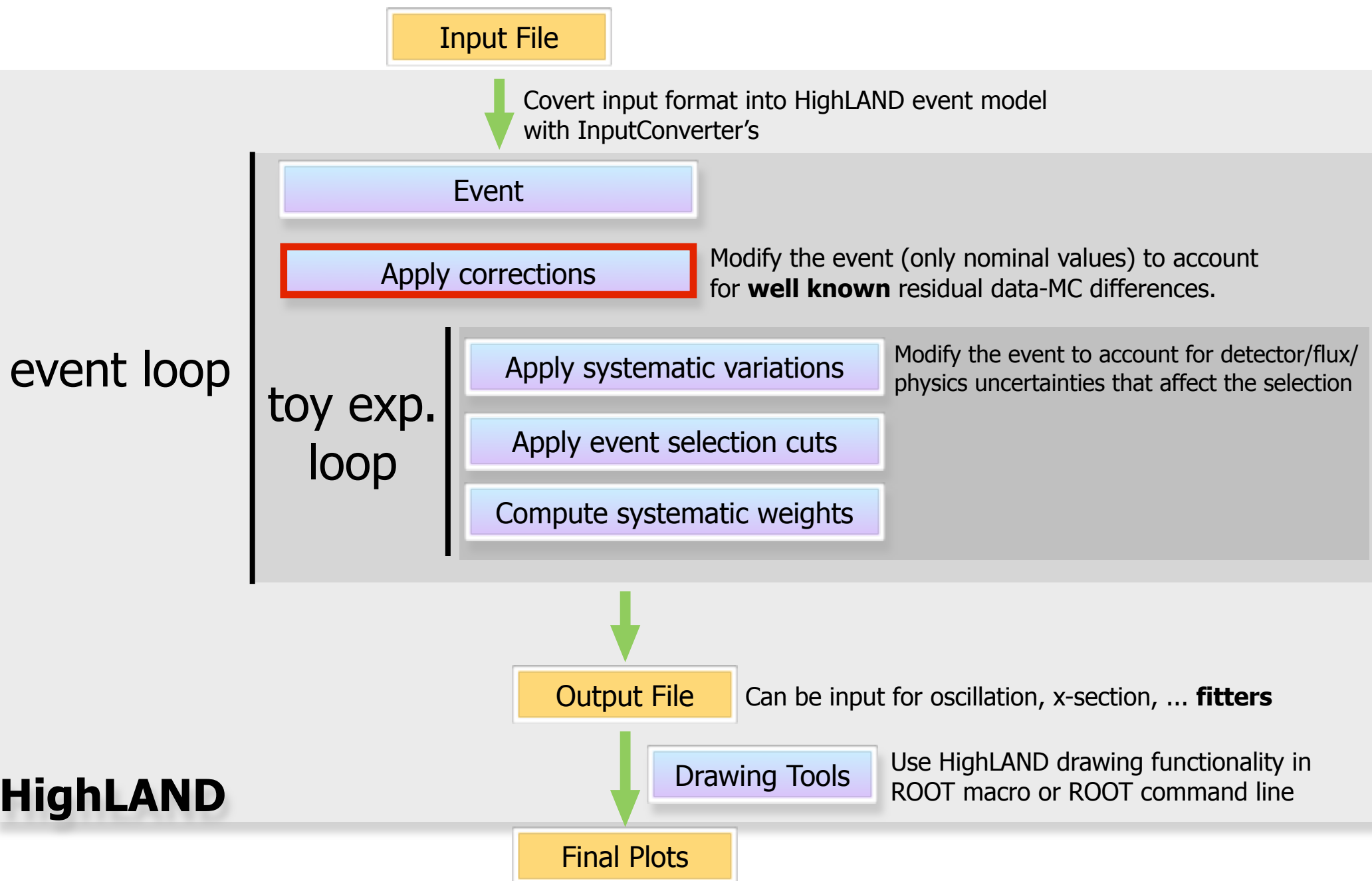
- In general we would need a data reduction process
  - i.e. Don't use hits at analysis level
- In T2K the output of the reconstruction is given to an intermediate package which filters (remove most hits, perform true-reco association, etc) the information and creates a new ROOT file ready for analysis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

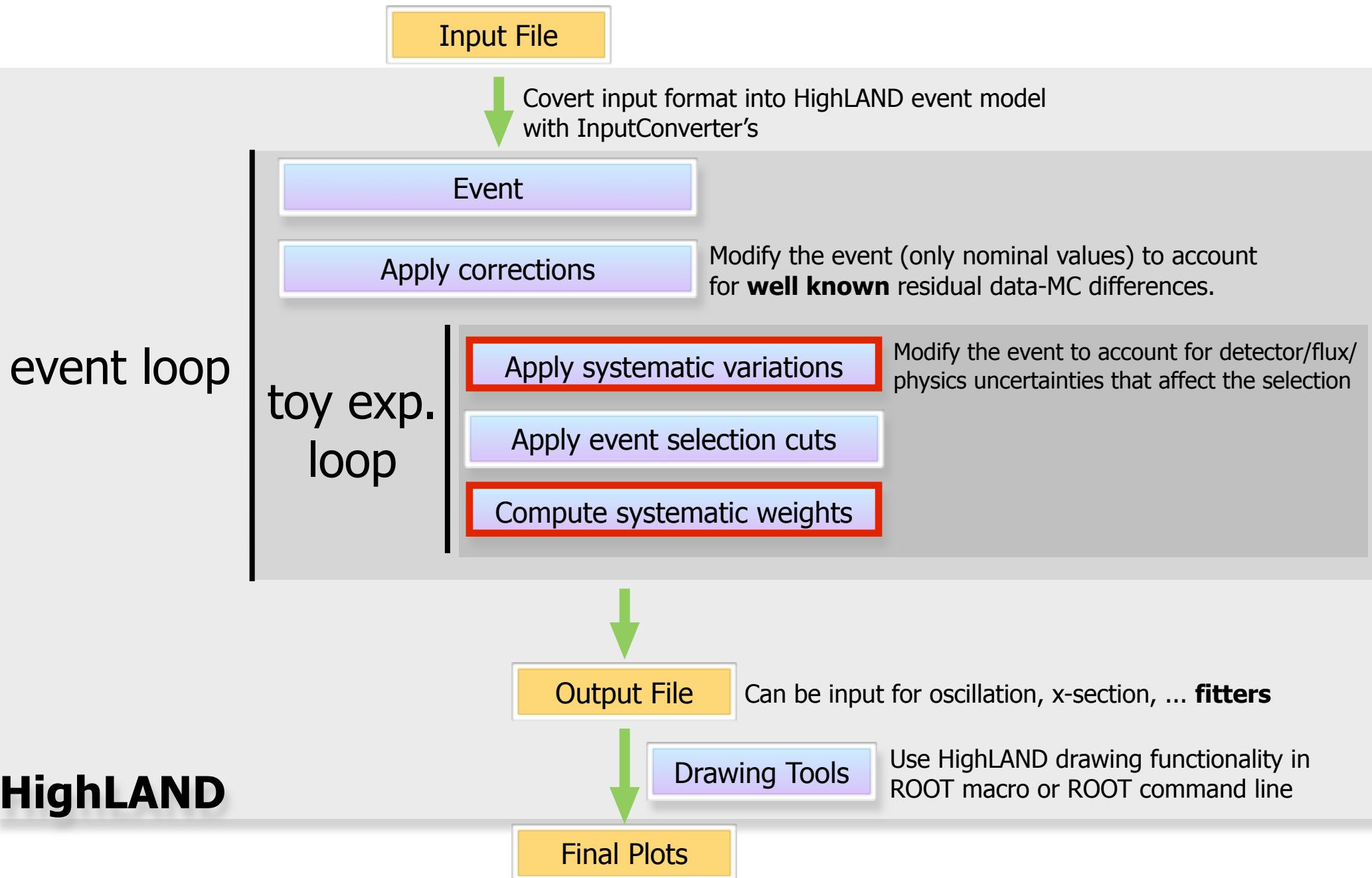


# Detector/reco optimization

---

- 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

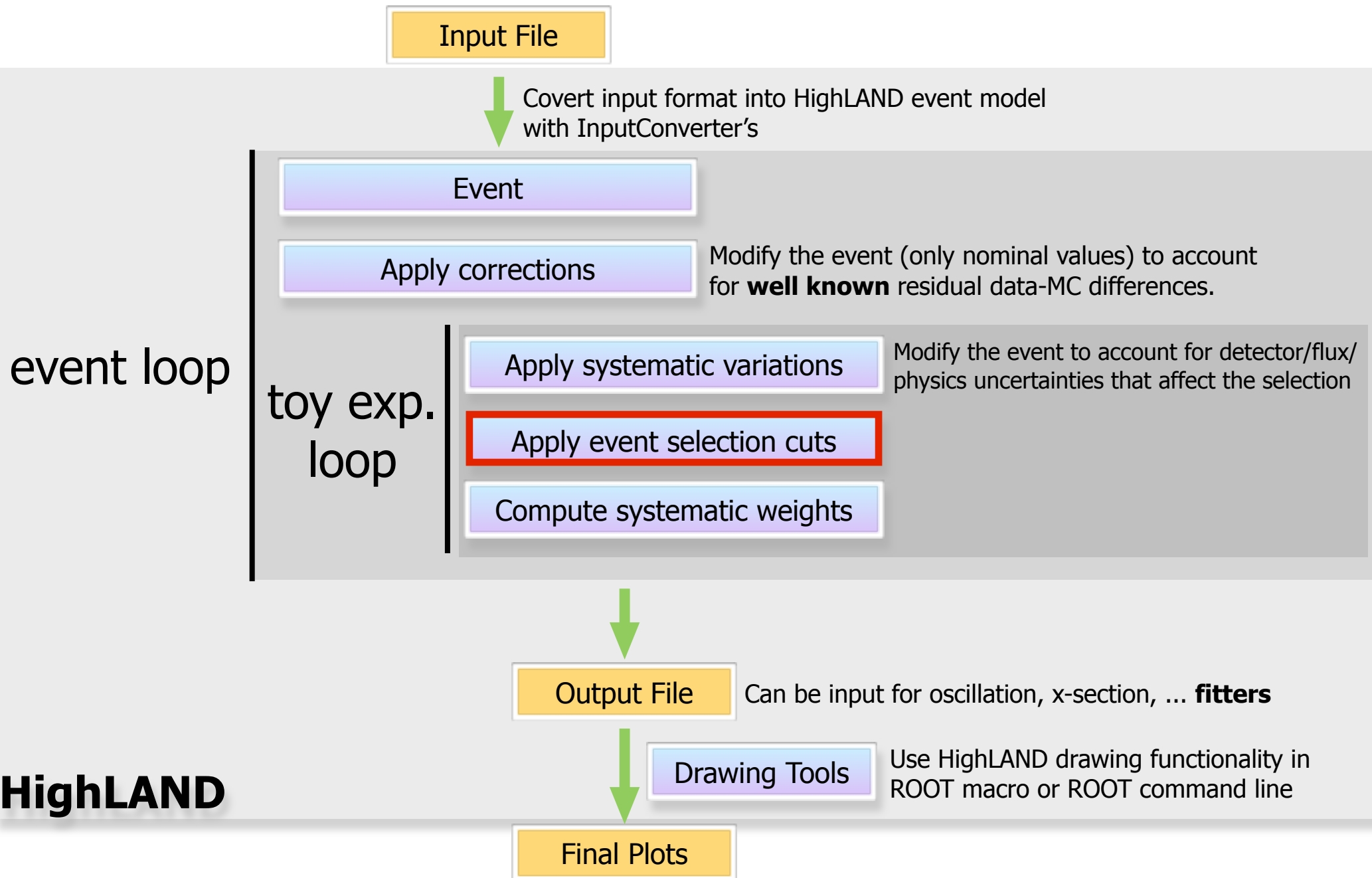
---

- 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;

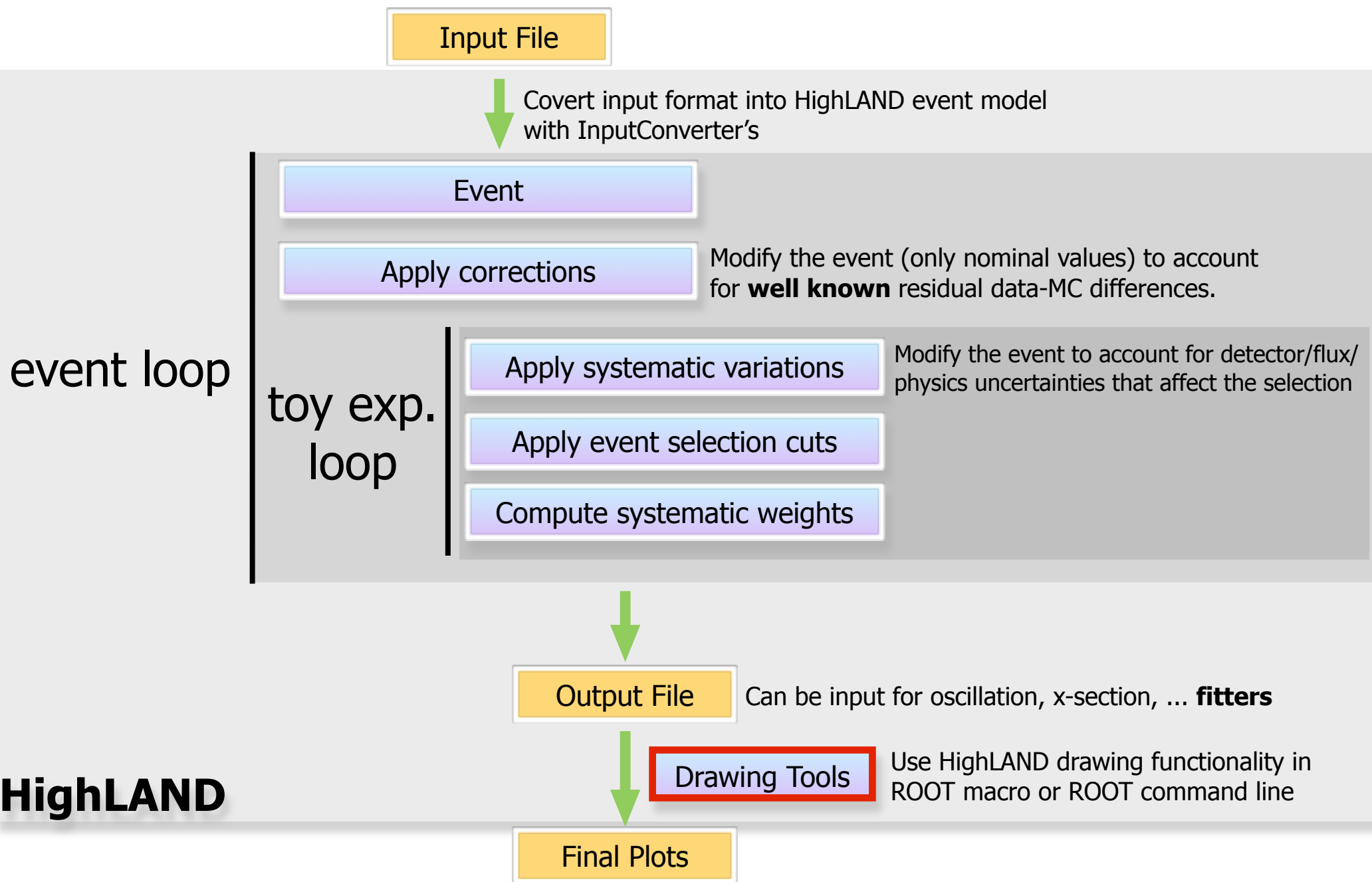
    // Apply the PID cut in the utilities namespace to the LeptonCandidate
    return cutUtils::MuonPIDCut( box.LeptonCandidate );
}

```

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



# Analysis flow



# 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

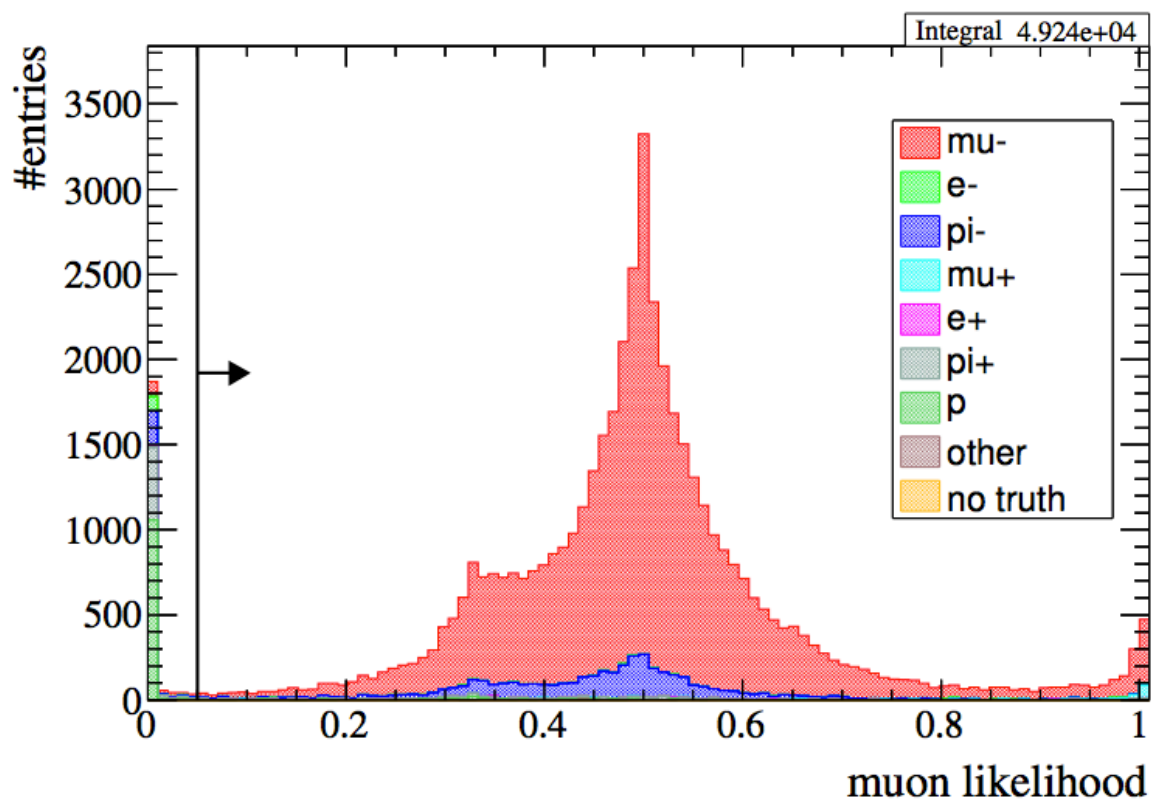
# Distributions

- 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")

```



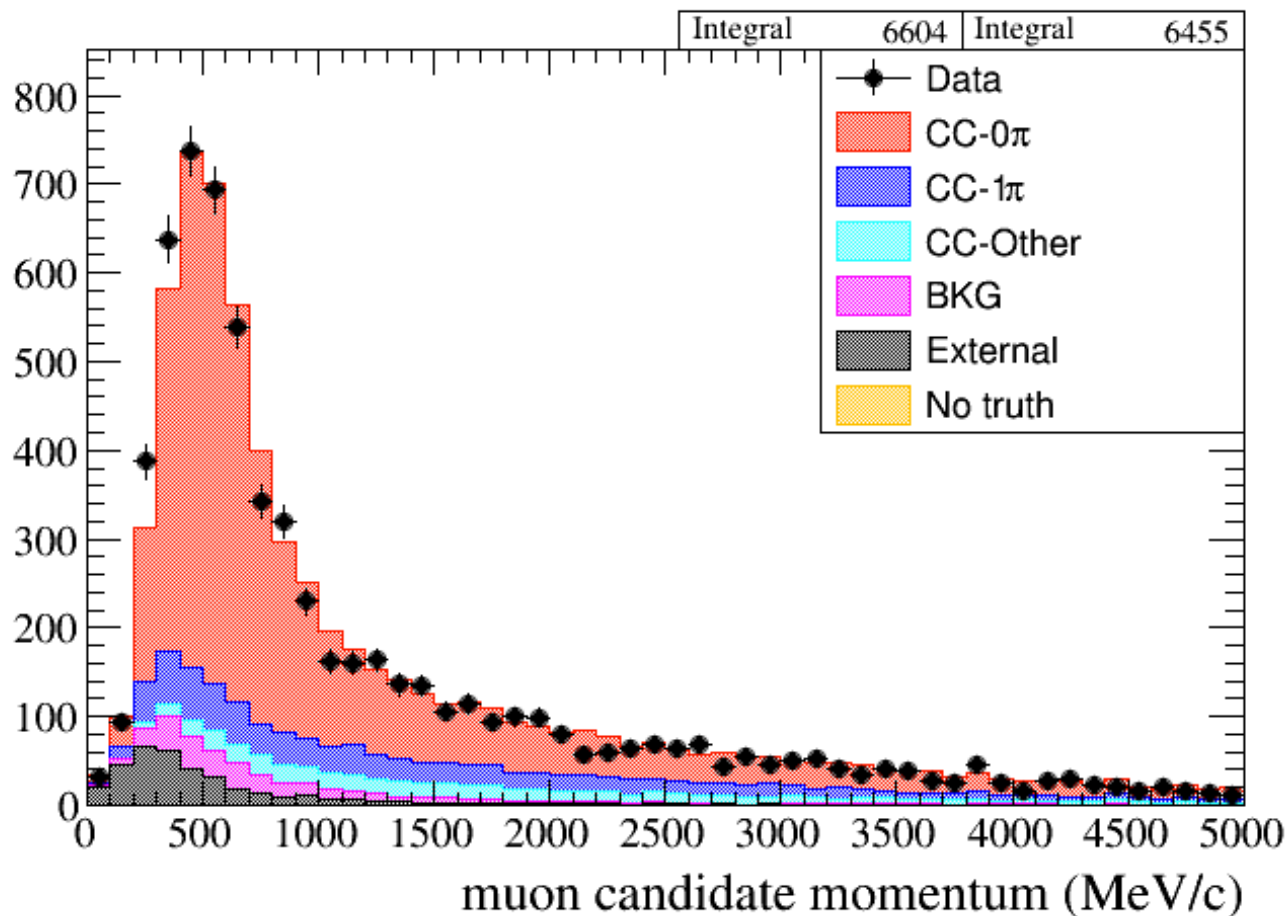
# Data/MC comparisons

- 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")

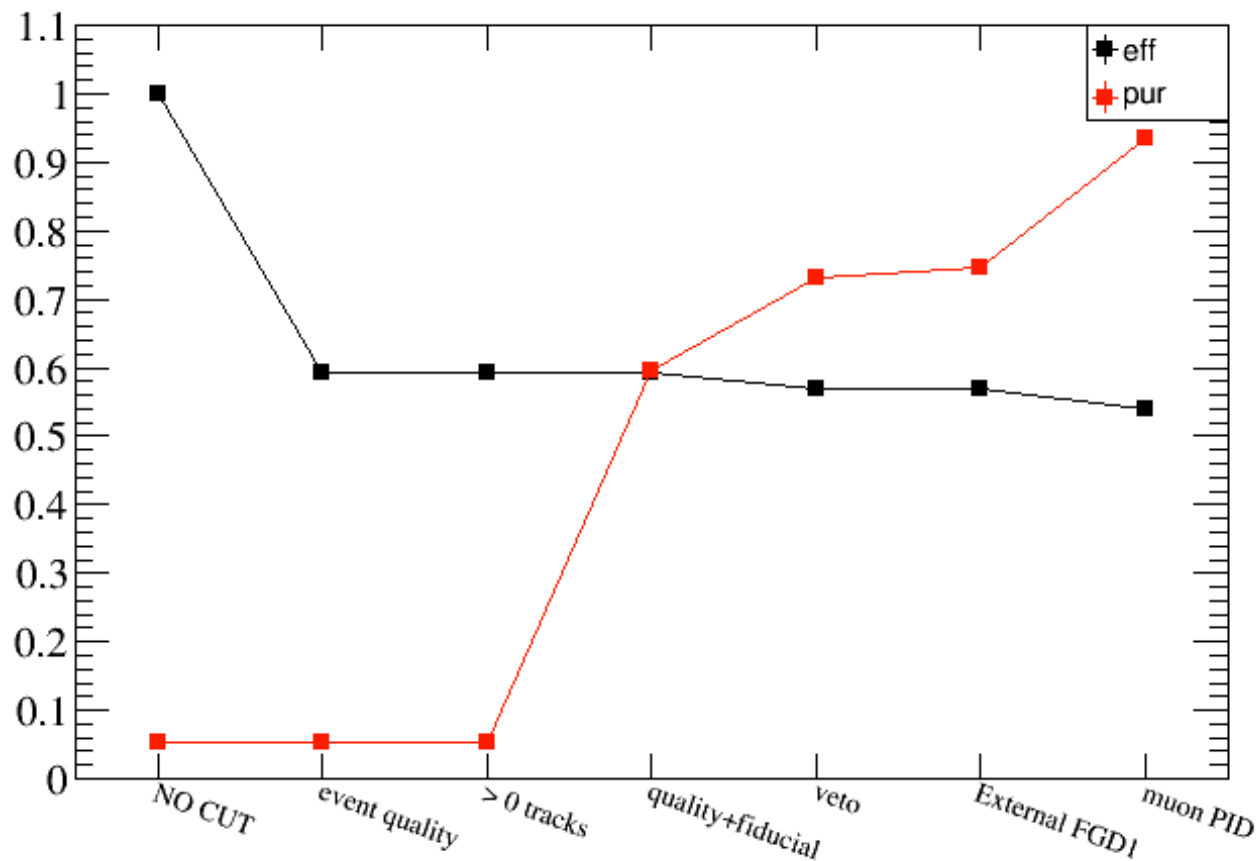
```



# Efficiencies & purities

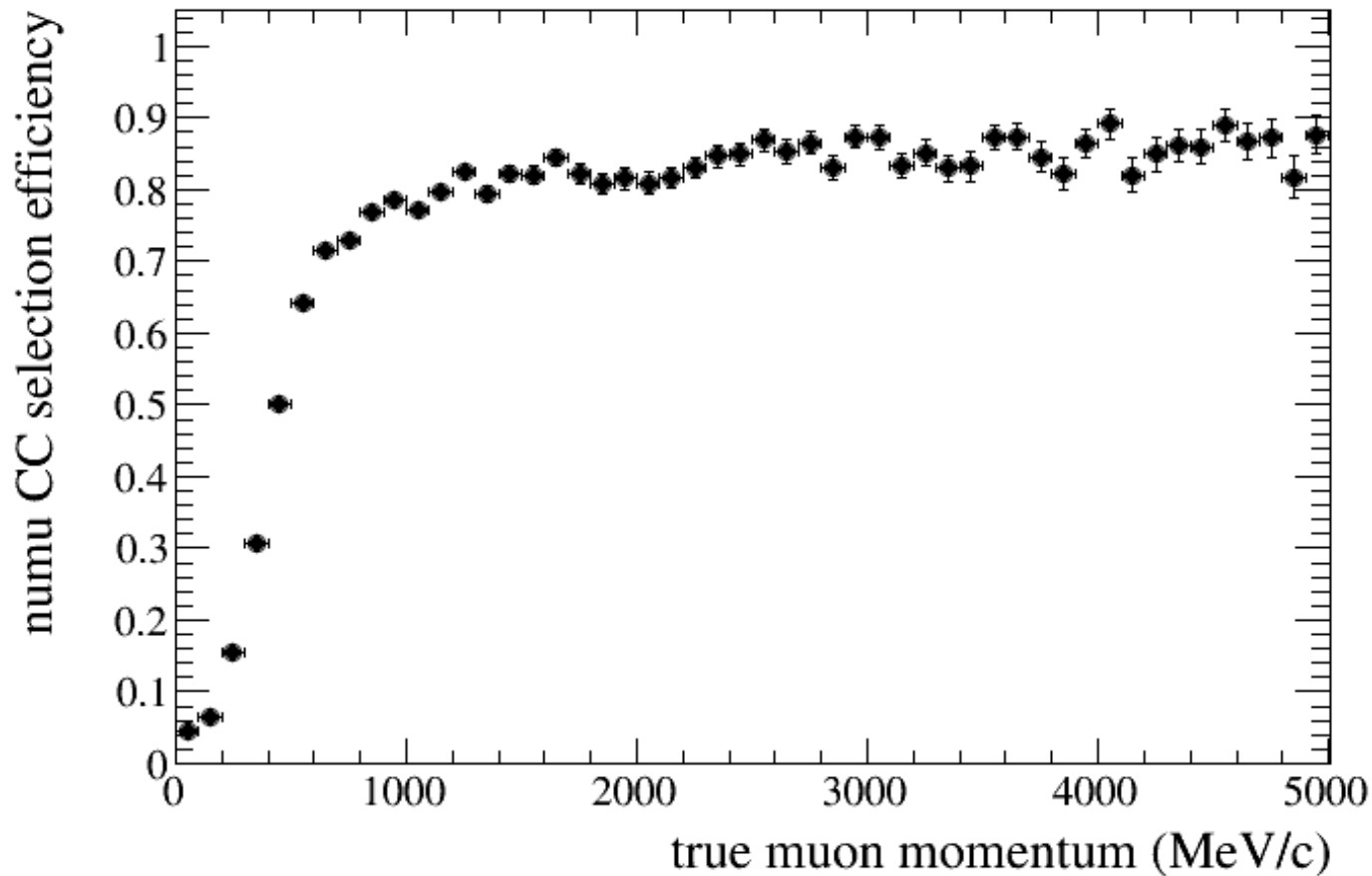
- Efficiency and purity after each cut in the selection

```
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");

```

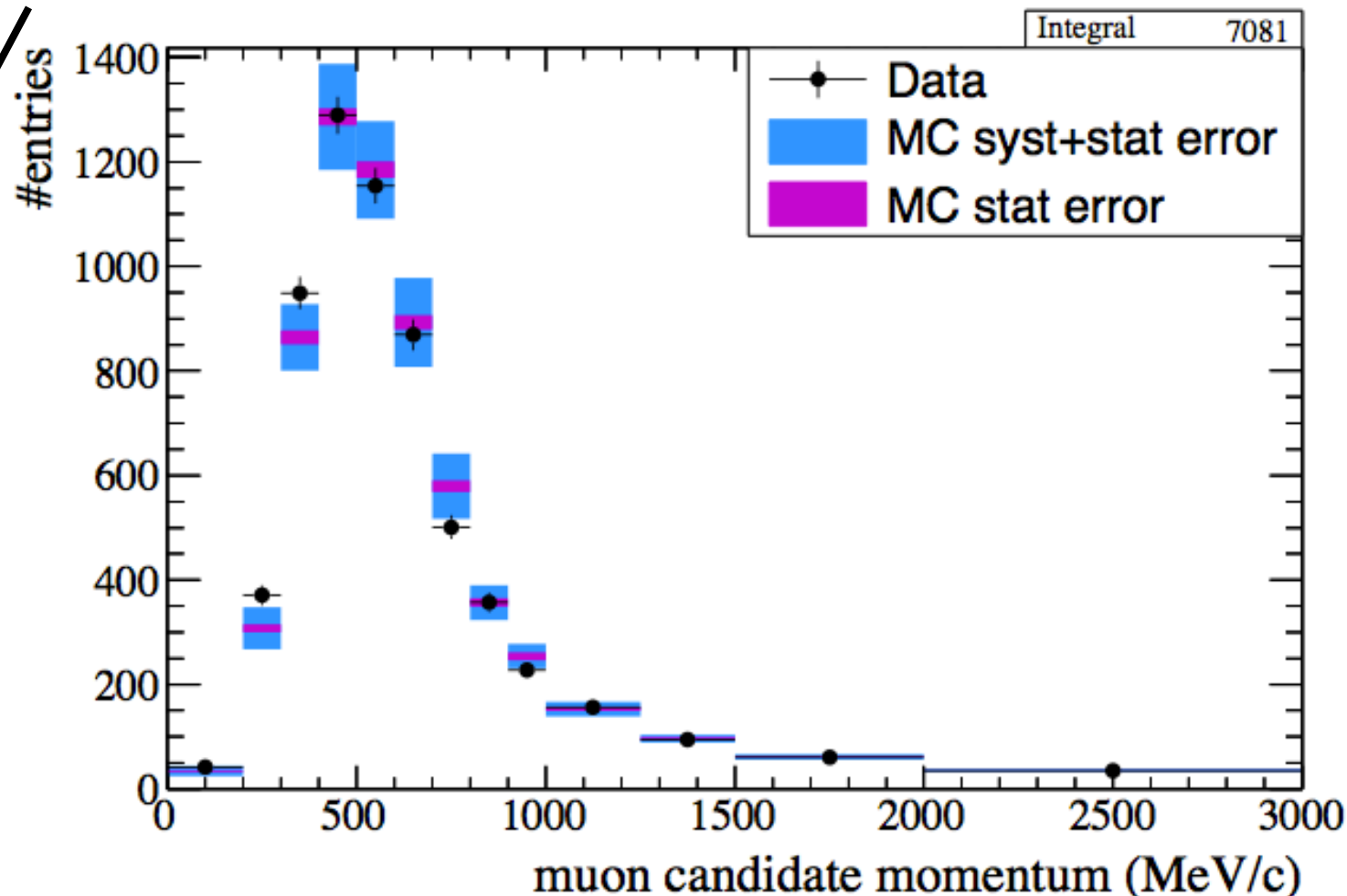
draw systematic

error bars

error style for MC

using  
Experiment class

variable  
binning

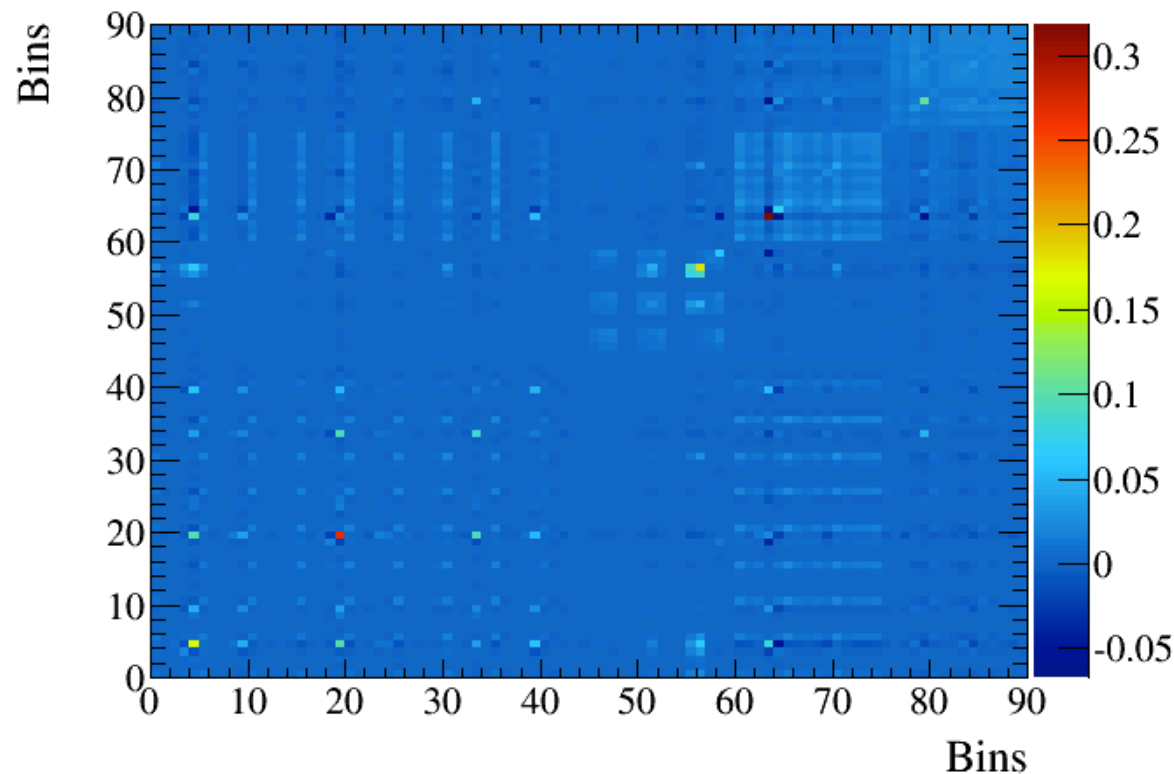




# Covariance Matrix

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