

vPRISM Software tutorial

Mark Scott
2nd vPRISM workshop
July 23rd 2014

Overview and goals

- Current software status:
 - What MC do we have?
 - What tools are there to do analyses?
 - Where can you get these?
- Goals of tutorial:
 - Download the software and files from the grid
 - Perform a flux coefficient fit
 - Apply reconstruction smearing and efficiencies
 - Apply flux and cross section weights
 - Produce an SK spectrum prediction

Downloading the software

- nuPRISM software stored in the T2K repository:
 - `export CVSROOT=:ext:anoncvs@repo.nd280.org:/home/trt2kmgr/T2KRepository`
 - `export CVS_RSH=ssh`
 - `unset CVS_SERVER`
 - `cvs co GlobalAnalysisTools/nuPRISM`
 - `cd GlobalAnalysisTools/nuPRISM`
 - `make links`
 - `make`
 - `source setup.sh`
- Setup.sh may complain if not using SL6:
 - Change line 2 from: `SCRIPT_PATH="`dirname \"$0`\""`
 - To: `SCRIPT_PATH="`dirname $0`"`

Downloading the files

- Some files are on the grid:
 - /grid/t2k.org/nd280/contrib/nuprism/
 - We will register the remaining files with the LFC and copy them to Europe in the coming weeks
- Today's files:
 - lcg-cp
lfn:/grid/t2k.org/nd280/contrib/nuprism/neut_5142/nuprism_numode_1km
/filename.root ./filename.root
- Or, from http://trshare.triumf.ca/~mscott/nuPRISM_Files/

nuPRISM files

- So far we have NEUT version 5.1.4.2 and 5.3.2 files
 - genev_320a_1km_nd2_1xx_17227.root
- Prefix tells you what type of file it is:
 - Genev = neutrino event file
 - Fluxweight = neutrino flux throw file
 - xsec_var = cross section throw file
- 320 = horn current, a = forward polarity, m = negative polarity
- 1km = distance of neutrino flux plane from neutrino target
- nd2 = off-axis flux plane
 - nd2 = from 0.8 degrees off-axis to 2.1 degrees
 - nd3 = 2.1 degrees to 3.4 degrees
 - nd4 = 3.4 to 4.7 degrees off-axis
- The rest is a run number, nxx, and a random ID

- Main classes/directories:
 - Processing nuPRISM and SK events
 - NuPrismVector
 - SKNtuple
 - fitQunEfficiency
 - Performing flux fits
 - Prob3++.20121225
 - Macro
 - Folders with data tables
 - eff_tables
 - Inputs
 - sk_flux_weights
 - sk_splines
 - src
 - The analysis code

nuPrismVector

- Class to read in genev, fluxweight and xsec_var files:
 - `NuPrismVector nuVectors(fVectorFile, fFluxWeightFile, fXSecWeightFile, mec);`
 - `fXXXFile` = text file with list of nuprism files
 - `mec` = Flag to turn on Nieves (`mec = 1`) or Martini (`mec = 2`) events
 - Your file lists should include files with MEC interactions
 - Not going to cover here, but feel free to ask me questions about the MEC side of things
- Iterates through nuPRISM events
- Provides event selections
- Access to systematic weights
- Access to reconstructed event properties

- Class to read in SK ntuple and xsec_var files:
 - `SKNtuple skMC(fSKVectorFile, fSKFluxWeightFile, fSKXSecWeightFile, mec);`
 - `fXXXFile` = text file with list of SK files
 - `FSKFluxWeightFile` = path to ROOT file containing flux splines
 - `mec` = Flag to turn on Nieves (`mec = 1`) or Martini (`mec = 2`) events
- Iterates through SK events
- Provides event selections
- Access to systematic weights
- Access to reconstructed event properties

fiTQunEfficiency

- Class to mock-up fiTQun reconstruction
 - `fiTQunEff = new fiTQunEfficiency(fEffFile,fResFile,19853423);`
 - `fEffFile` = fiTQun reconstruction efficiency table file
 - `fResFile` = fiTQun reconstruction smearing file
 - Random seed – should be kept constant
- Main methods are shown below:

```
fiTQunEff->FindMERandTopology(nuVectors);  
fiTQunEff->CalculateToWall(nuVectors);  
fiTQunEff->CalculateEfficiency(nuVectors);  
// Get fiTQun reconstruction efficiency  
double eff = nuVectors.fiTQunEff[1];  
// Apply efficiency (throw random events away)  
if(rand.Rndm()>eff) continue;  
// Apply the smearing to give reconstructed  
quantities  
fiTQunEff->ApplySmearing(nuVectors);
```

- Oscillation probabilities
 - We use the probNuPRISM.h class
 - `ProbNuPrism* osc_prob = new ProbNuPrism(dm2, theta23);`
- Two initialisation methods:
 - `ProbNuPrism(double Delta_M2, double theta23, double delta_m2, double theta13, double theta12, double Delta);`
 - `ProbNuPrism(double Delta_M2 = 2.4e-3, double theta23 = 0.5);`
- Set whether we're looking at anti-neutrino oscillations and which mass hierarchy:
 - `void ProbNuPrism::SetAntiNeutrino(bool isAntiNeutrino)`
 - `void ProbNuPrism::SetInverseMassHierarchy(bool isInverted)`
- Methods to get any oscillation probability:
 - `double ProbNuPrism::GetProbNuMuNuMu(float energy)`

macros

- Directory of ROOT macros to perform fits of the nuPRISM flux
 - `fit_spectrum_SKNue.cc`
 - `fit_spectrum_BeamNue.cc`
 - `fit_spectrum_WrongSign.cc`
 - `fit_spectrum_numu_disappearance.cc`
- Most macros just about work – need more validation
- Can use the `compile_fit_spectrum_Prob3.cc` macro to load the necessary libraries for the fit macro

macros

- Directory of ROOT macros to perform fits of the nuPRISM flux
 - `fit_spectrum_SKNue.cc`
 - `fit_spectrum_BeamNue.cc`
 - `fit_spectrum_WrongSign.cc`
 - `fit_spectrum_numu_disappearance.cc`
- Most macros just about work – need more validation
- Can use the `compile_fit_spectrum_Prob3.cc` macro to load the necessary libraries for the fit macro

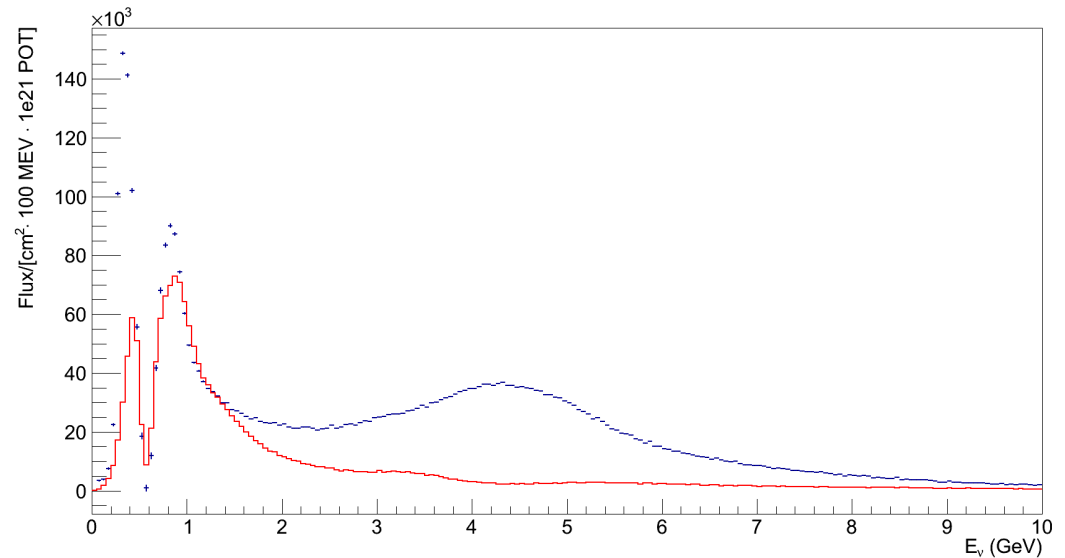
First task!

- Modify `compile_fit_spectrum_Prob3.cc` to load the `fit_spectrum_numu_disappearance.cc` macro
- Load the macro:
 - `root -l compile_fit_spectrum_Prob3.cc`
- `fit_spectrum(double theta, double dm2, bool appearance = false, double elo = 0.4, double ehi = 3.0)`
 - `theta` = value of $\sin^2(\theta_{23})$
 - `dm2` = value of Δm^2_{23}
 - `appearance` = should always be false, not tested
 - `elo` = lowest energy to fit
 - `ehi` = highest energy to fit

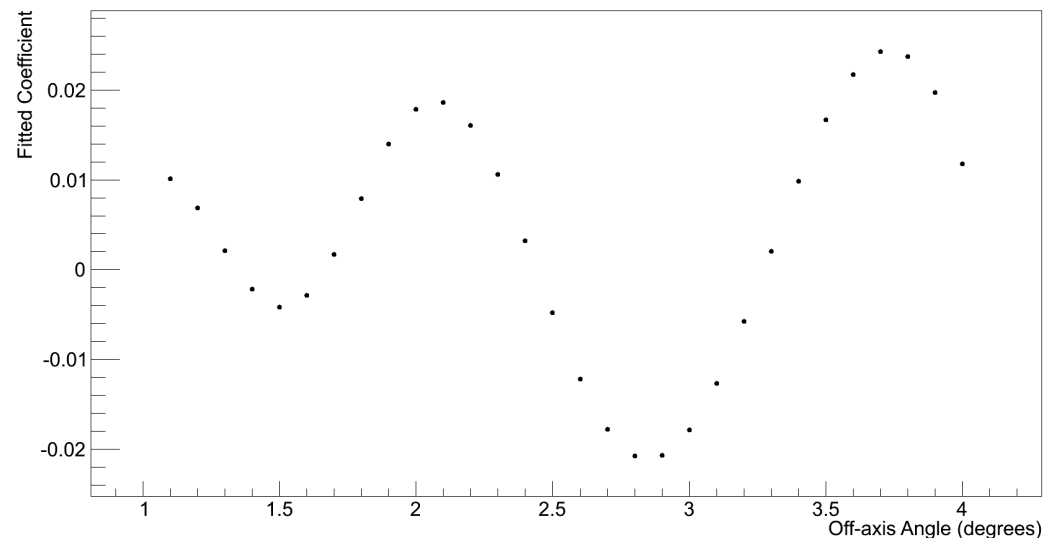
First task!

- Try running: `fit_spectrum(0.5, 0.00241, 0, 0.4, 1.5)`

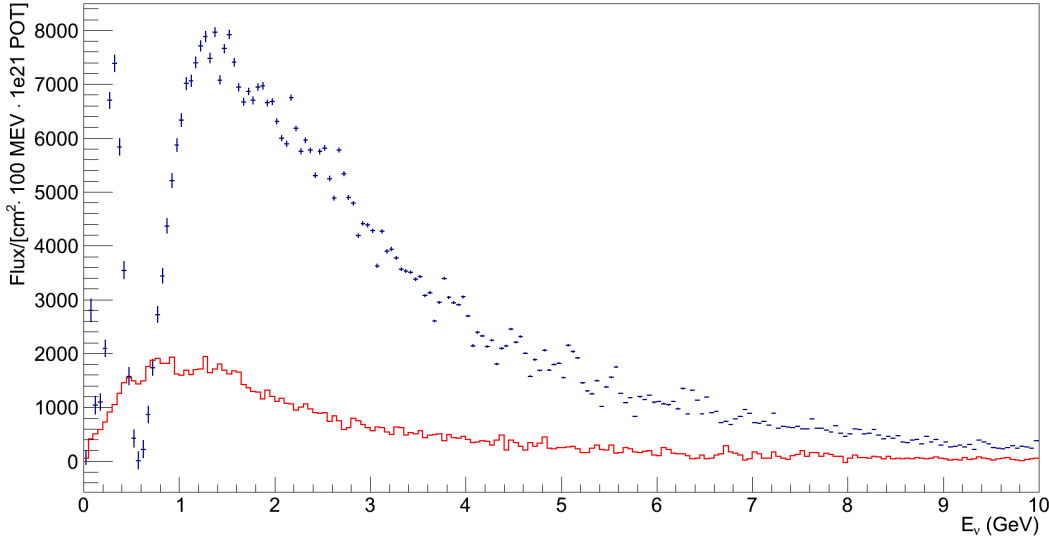
- Look at canvas 1 – the nuPRISM numu fit (red) to the oscillated SK numu spectrum (blue)



- Canvas 2 shows the fit coefficients as a function of off-axis angle



First task!

- Other histograms less good... show numu_bar, c4, and nue, c6, flux fits using numu coefficients
 - c3 is the fractional difference between the nuPRISM numu fitted flux and the SK numu flux
 - c5 is the same for the cumu_bar fluxes
- 
- Produces an output file:
 - `nuprism_coef_newsmooth_241_50000.root`
 - Contains the fit coefficients
 - Input to the nuPRISM analysis

- Focus on `nuprism_numu_disappearance_analysis.cc`
 - It uses all the steps discussed previously
- Executable: `bin/nuprism_numu_disappearance_analysis`
- Options:
 - f = text file with list of nuPRISM genev files
 - w = text file with list of nuPRISM fluxweight files
 - x = text file with list of nuPRISM xsec_var files
 - skf = text file with list of SK event files
 - skw = Path to the `sk_flux_weights/sk_weights_numode.root`
 - skx = text file containing the SK xsec_var files
 - e = Path to `eff_tables/fQEffTbl.root`
 - r = Path to `eff_tables/fQSmrTbl.root`
 - c = Path to coefficient file
 - o = Output filename.root
 - dm2 = value of Δm^2_{23}
 - theta23 = value of $\sin^2(\theta_{23})$
 - mec = Are we adding Nieves (mec = 1) or Martini (mec = 2) MEC events

nuprism_numu disappearance_analysis

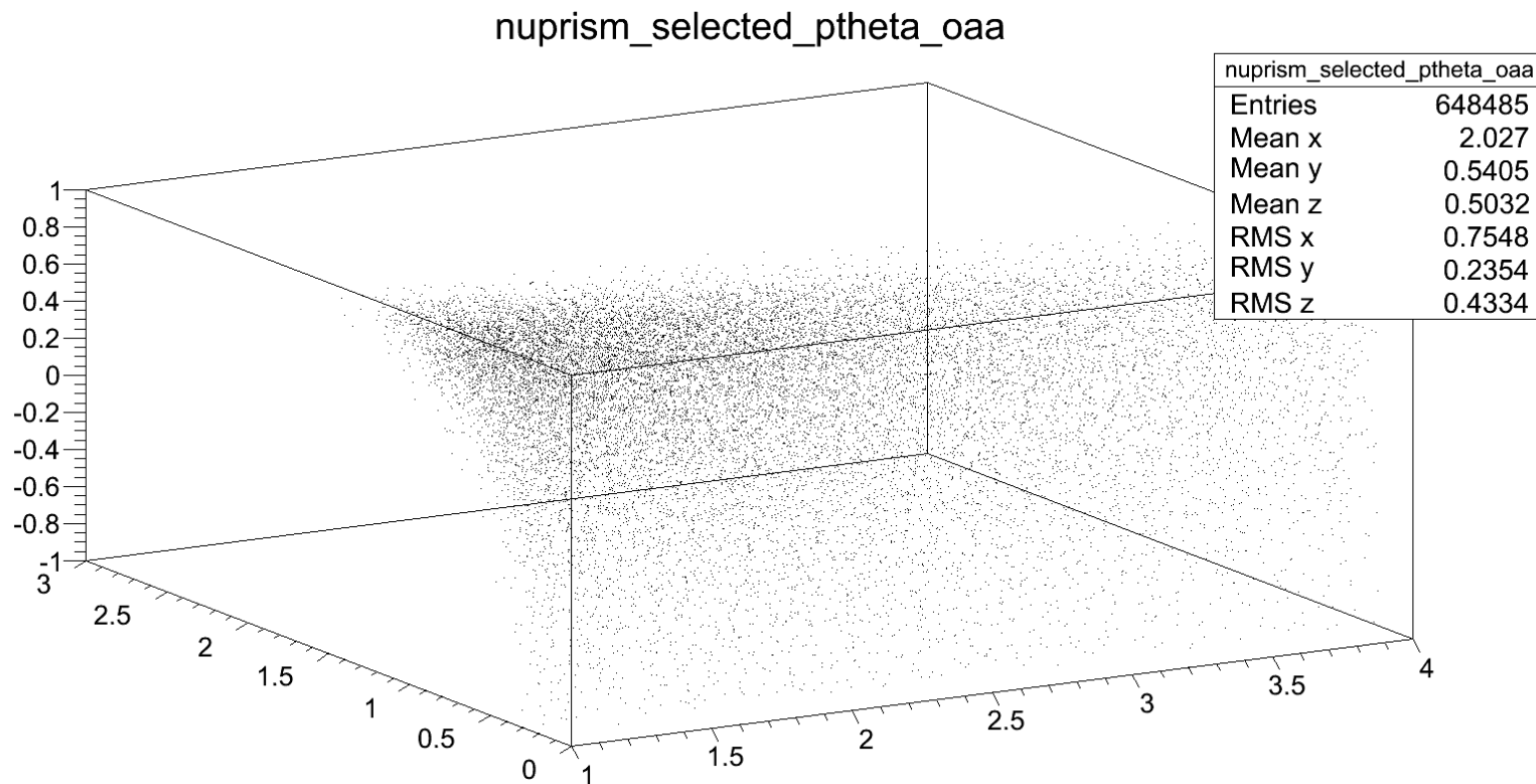
- Selects nuPRISM events
- Applies analysis corrections
- Calculates the predicted number of events at SK
- Also applies cross section and flux uncertainty throws
- Calculates systematic covariance matrix
- Calculates statistical uncertainty matrix

- List the nuPRISM and SK files you downloaded in the appropriate text file:
 - `ls /path/to/genev/files*.root >> genev_nuprism.txt`
 - `ls /path/to/fluxweight/files*.root >> fluxweight_nuprism.txt`
 - `ls /path/to/xsec_var/files*.root >> xsec_nuprism.txt`
 - `ls /path/to/sk/vector/files*.root >> sk_events.txt`
 - `ls /path/to/sk/xsec_var/files*.root >> xsec_sk_events.txt`
- From the top nuPRISM directory try running:

```
./bin/nuprism_numu_disappearance_analysis -f genev_nuprism.txt -w  
fluxweight_nuprism.txt -x xsec_nuprism.txt -skf sk_events.txt -skw  
./sk_flux_weights/sk_weights_numode.root -skx xsec_sk_events.txt  
-e eff_tables/fQEffTbl.root -r eff_tables/fQSmrTbl.root -c  
macros/nuprism_coef_newsmooth_241_50000.root -o output.root  
-dm2 0.00241 -theta23 0.5 -mec 0
```

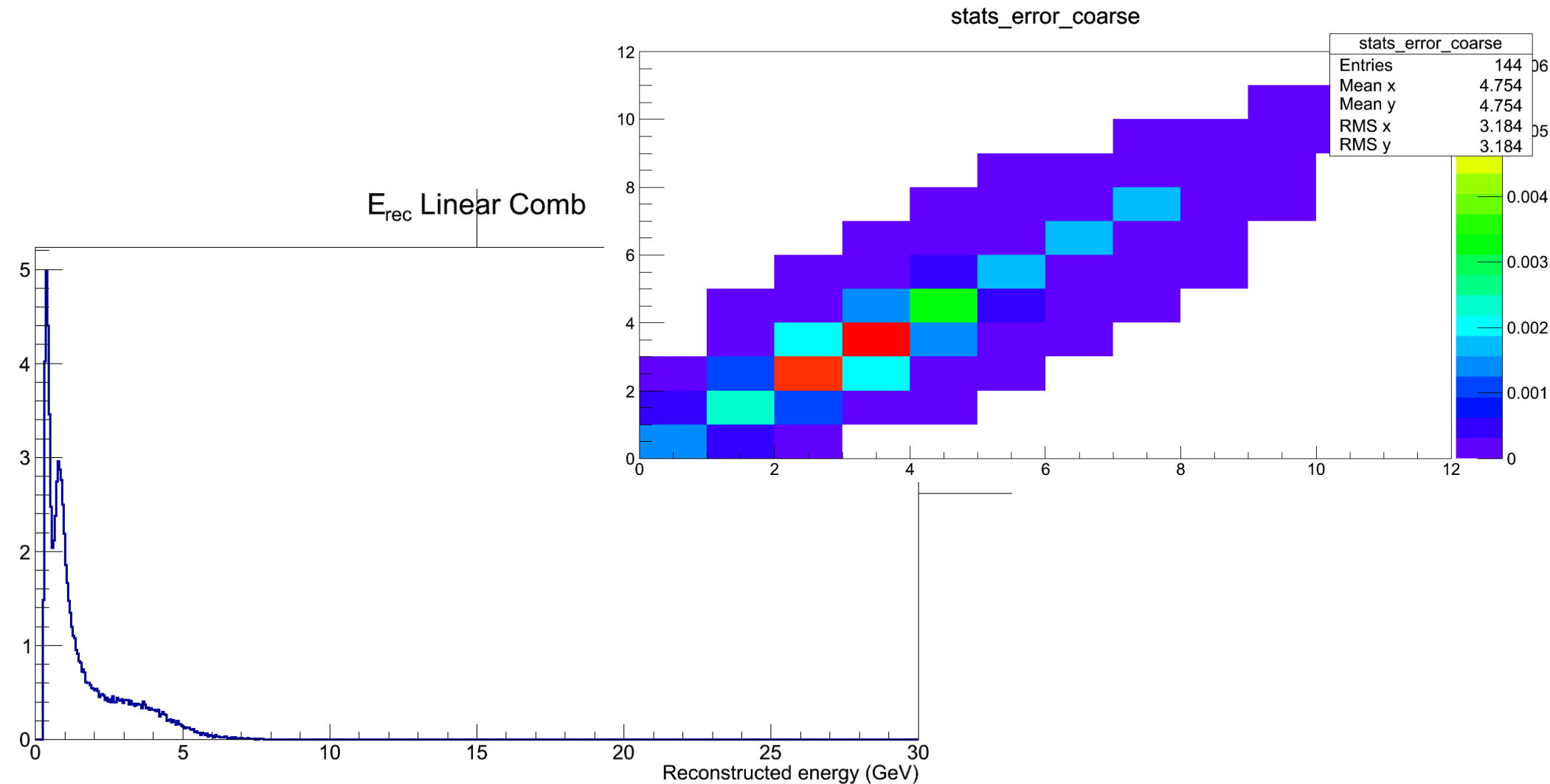
Analysis output 1

- Horrible file with terrible naming of histograms!
 - Apologies, this will be fixed soon
- Take a look at `nuprism_selected_ptheta_oaa` – the selected events at nuPRISM



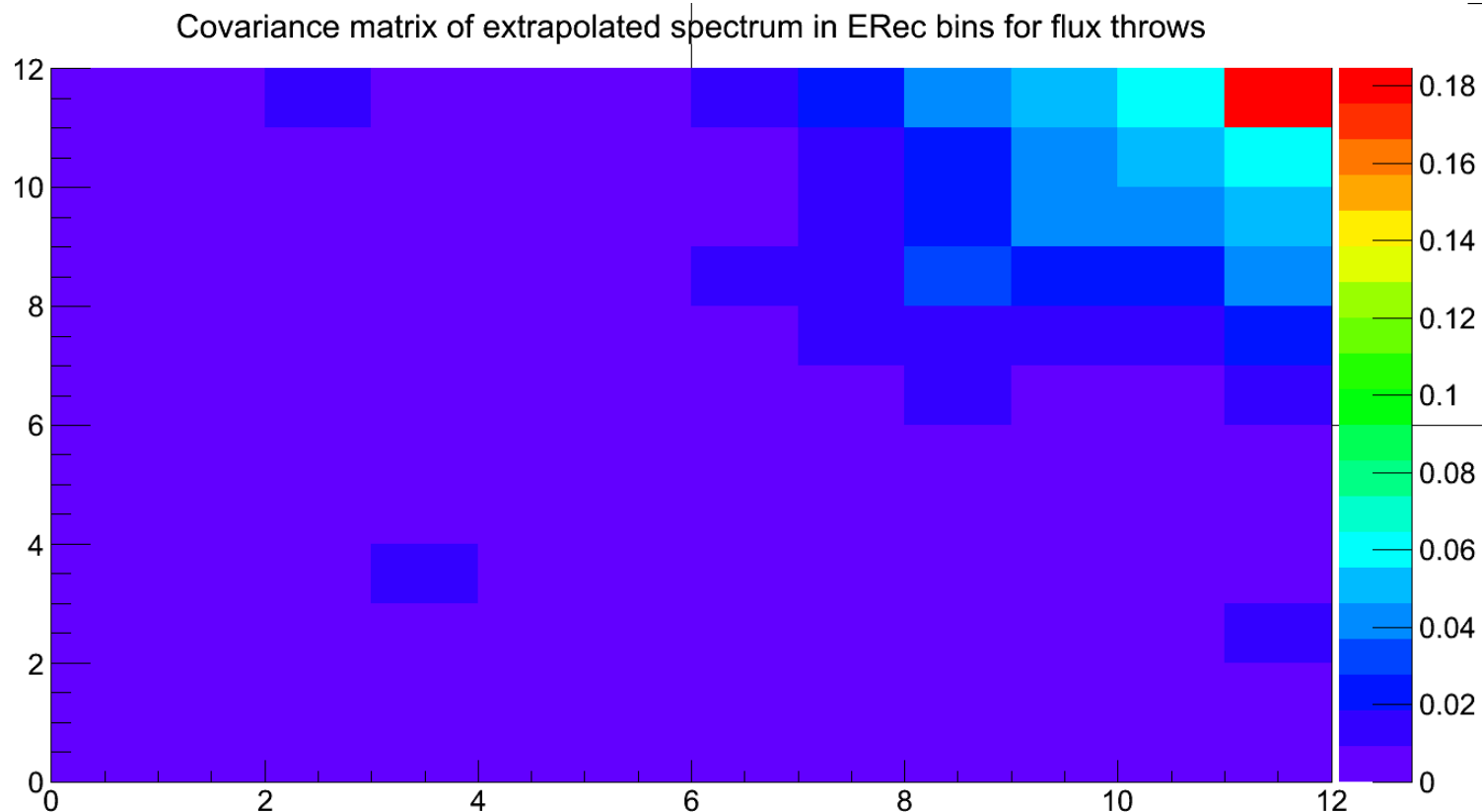
Analysis output 2

- stats_error_coarse stores the statistics covariance matrix
- erec_linear_comb is the nuPRISM prediction at SK



Analysis output 3

- Final thing, `erec_covariance_total` = final covariance matrix



- `erec_covariance_flux_0` = XSec covariance matrix
- `erec_covariance_flux_1` = Flux and XSec covariance
- `erec_covariance_flux_n` = Flux source (n-1) covariance

- Code and analysis is evolving rapidly
 - Have code to perform all the steps needed to do an oscillation analysis
 - Will be cleaned up and better commented in the near future
- Please feel free to email me or Mark Hartz if you have any questions or suggestions
 - Don't suffer in silence!