Preparing for CLAS12 Data Analysis

Derek Glazier
University of Glasgow
Hadron Spectroscopy Working Group
3/30/2017

With HASPECT collaboration :
Edinburgh, JPAC, Giessen, Glasgow, Genova, Mainz, Ohio, Torino, Roma, ...

Overview

- Full Analysis Chain
- Signal Selection
- Observable Extraction
- Extraction Validation
- hsroot coding environment

Experimental Data Analysis

Goal : Extract physics observables from experimental data distributions

Issues: To extract reliable results for subsequent theoretical interpretation we must

Account for acceptance

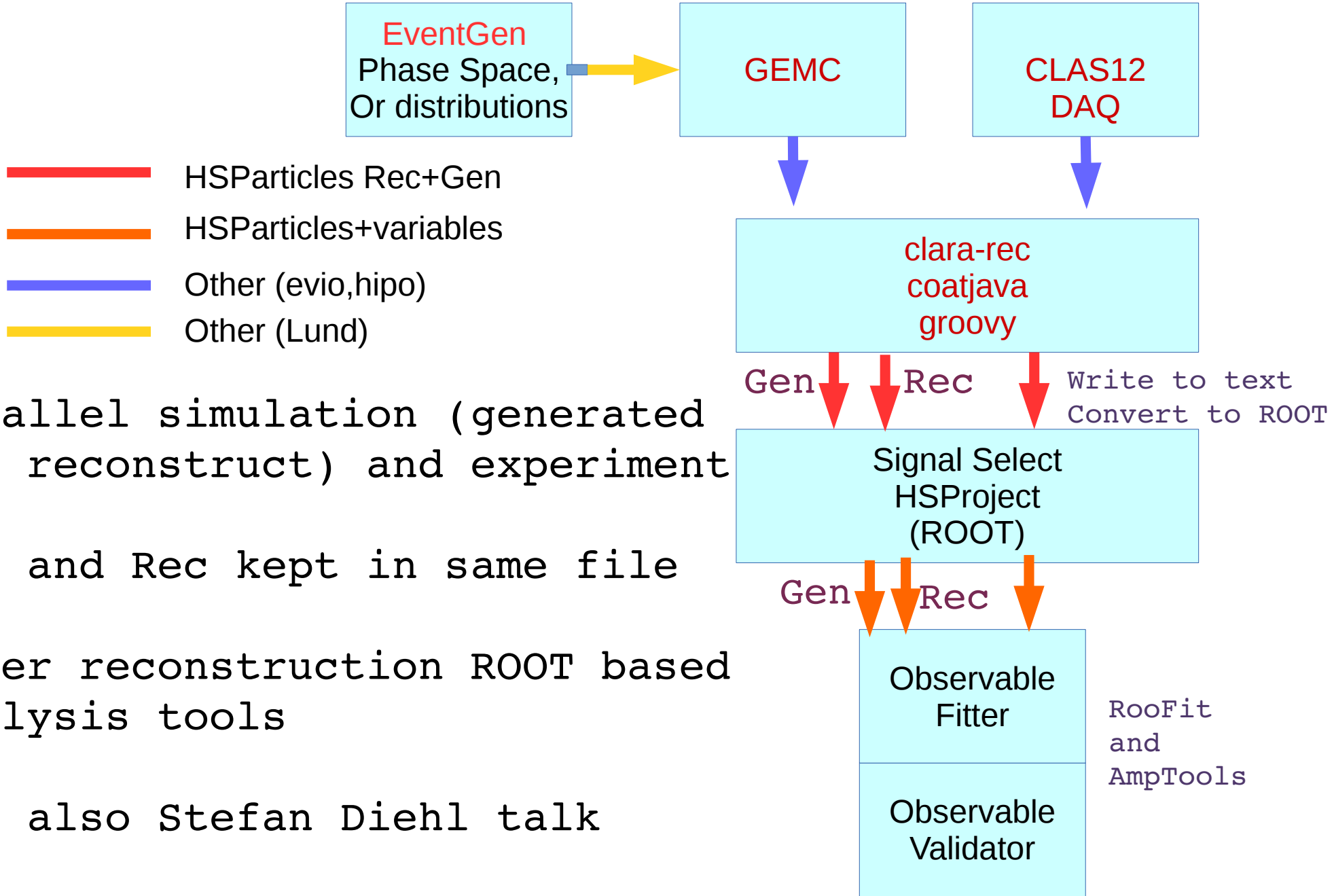
Account for resolutions

Account for backgrounds

...

+ Know how well we have handled these (systematic uncertainties)

HASPECT Data Analysis



Parallel simulation (generated and reconstruct) and experiment

Gen and Rec kept in same file

After reconstruction ROOT based analysis tools

See also Stefan Diehl talk

THSProject Class

- Handle reconstructed events output from coatjava
  - Contain vectors of HSParticles
  - Experiment, Generated and Reconstruct data analysed with same code
- Prepare variables for fitting
  - AmpTools, RooFit, …
- Programme different possible final states (topologies) to run simultaneously
  - Detected dependent and independent parts
- Permutates and creates events for like particles
- Flags if correct permutation recorded (algorithm dependent)
- Can filter PID and/or charge state
- Can select inclusive final states
- Can be used with ROOT TSelector/PROOF

THSParticle Class

- C++/ROOT based class
- Contains useful information
  - 4-vector, vertex, time, PDG, PID
  - If simulated also Truth values
- Similar to coatjava Particle classes

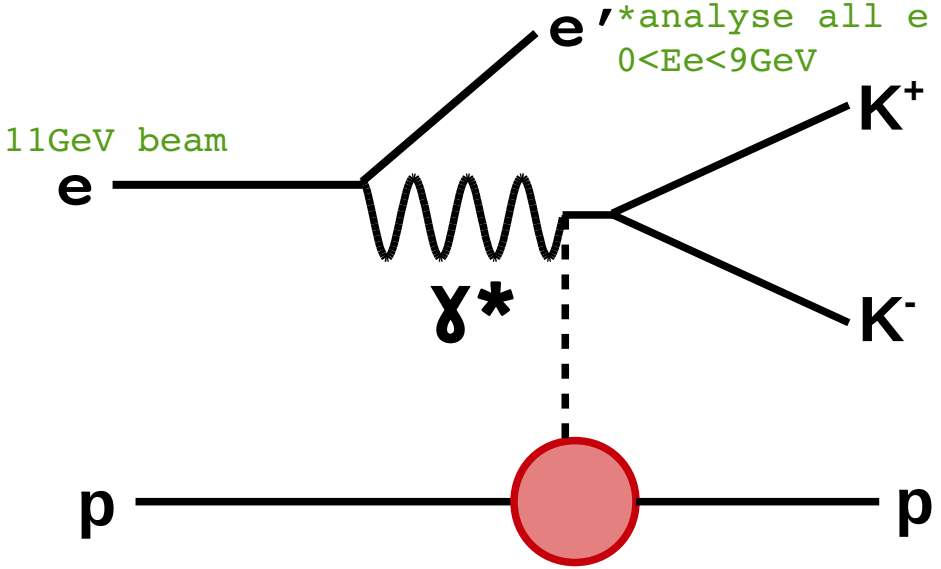
THSKinematics

- Kinematic calculator for :
- Photo and electro production
  - W,t,t-t0,Q2,polarisation…
- 2-body decays
  - S-channel CM
  - Helicity angles
  - GJ angles
- 3-body decays
  - Not yet...

e.g. γ+p → N*M*
N* → ωp and M* → π+π-

```
fKine.SetGammaTarget(fBeam,fTarget);
fKine.SetMesonBaryon(
      fPip.P4()+fPim.P4(),
      fProton.P4()+fOmega.P4());
f_W=fKine.W();
f_t=fKine.t(); //t for pi+pi-
                meson production
fKine.PhotoCMDecay();
fCMCosTh=fKine.CosTheta();
fCMPhi=fKine.Phi();
fKine.MesonDecayHelicity()
fHelCosTh=fKine.CosTheta();
fHelPhi=fKine.Phi();
```

Toy Example Full Chain Analysis



* FT reconstruction wasn't "on"
Used true e-' values
Simple to mix gen rec in THSProject

Simulate 40k phase space events with gemc

Reconstruct with coatjava

Write to text with groovy

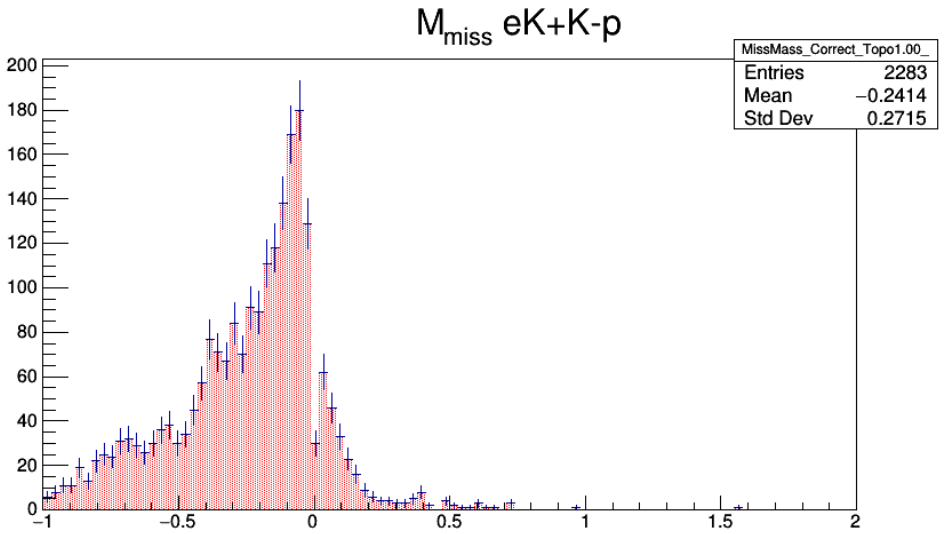
Convert to ROOT HSParticle

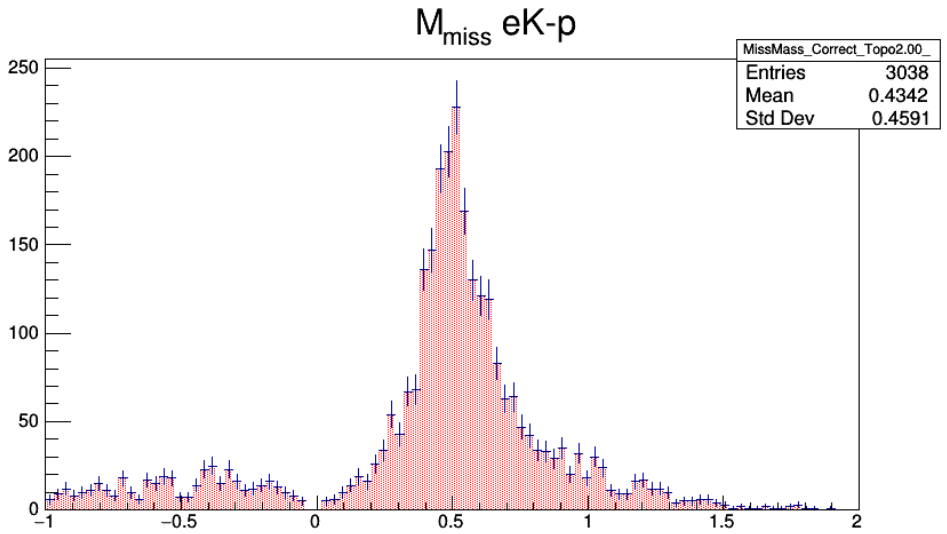
Analyse with THSProject

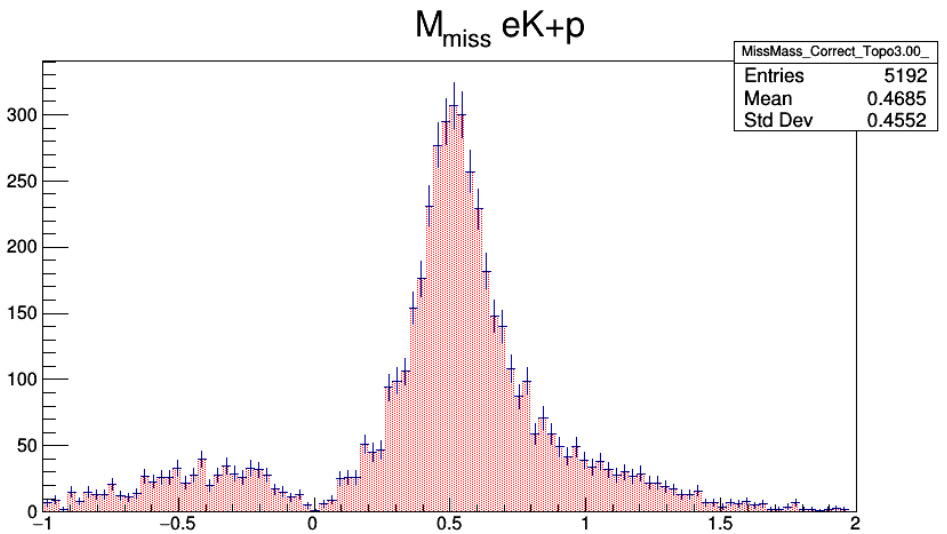
Reconstruct 4 Topologies

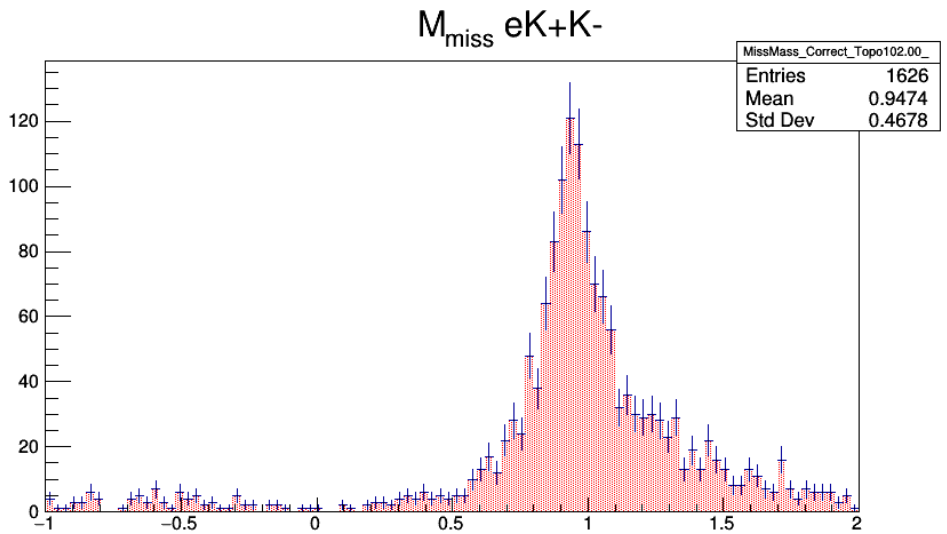
Using a THSProject class ID done on CHARGE state

Cut on "Correct" Permutation (only possible with sim.)

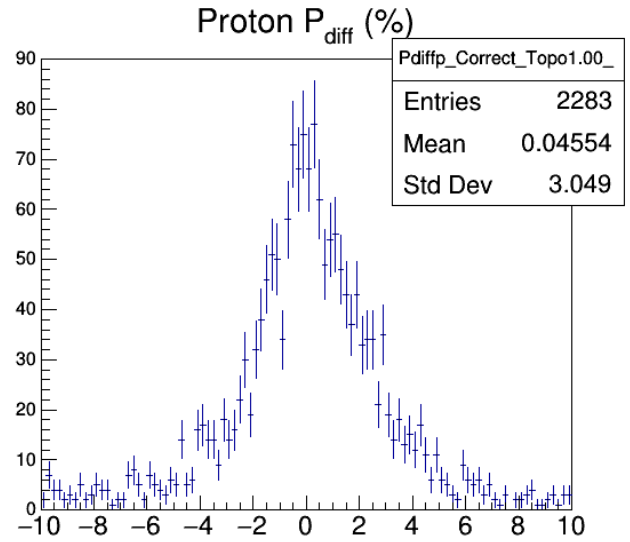


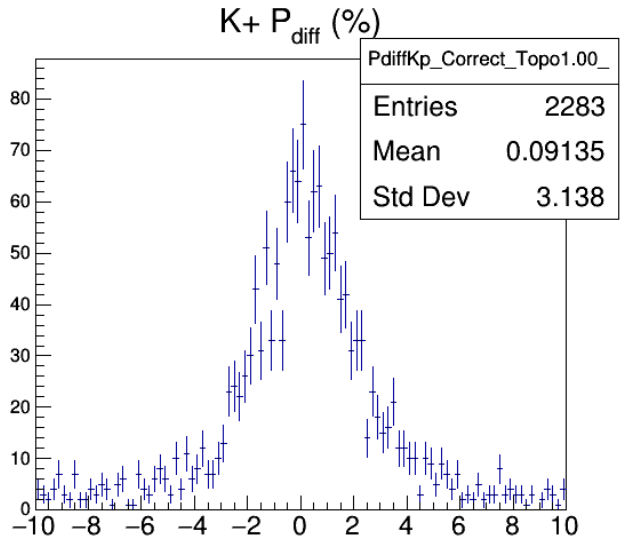


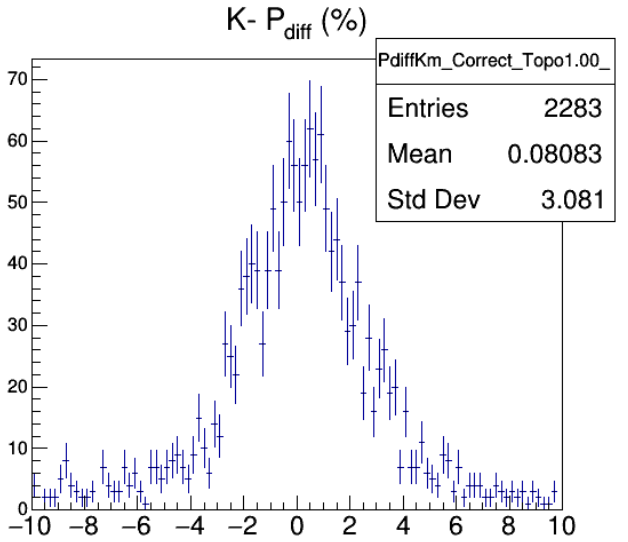


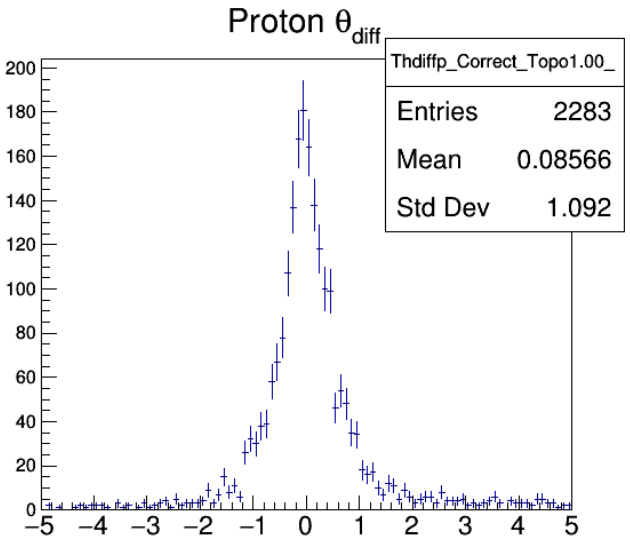


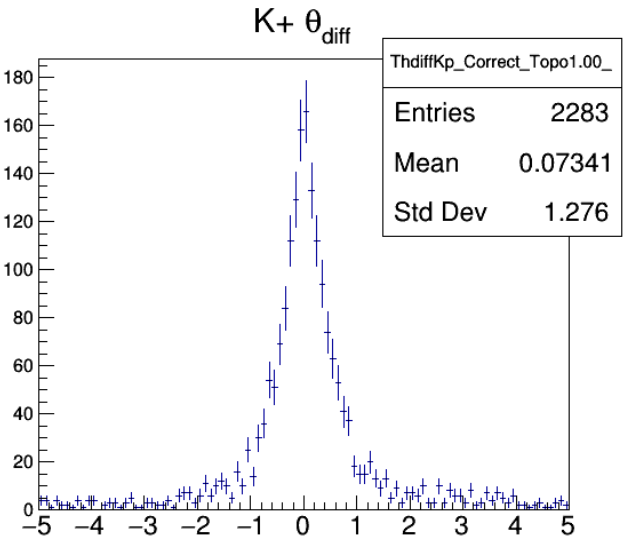
Detector Resolution Histograms

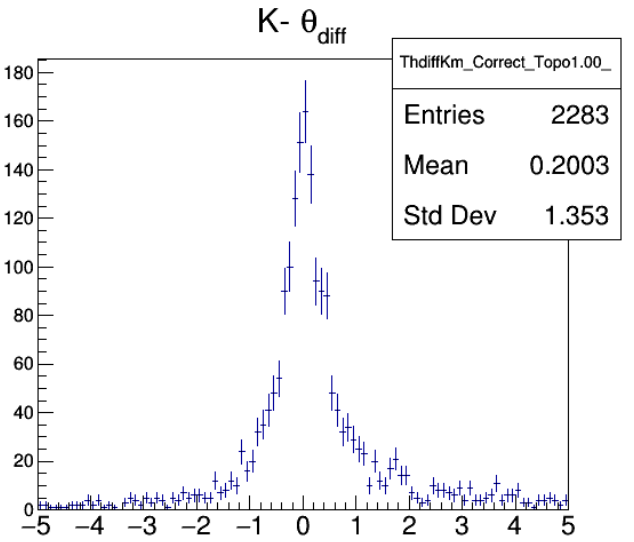






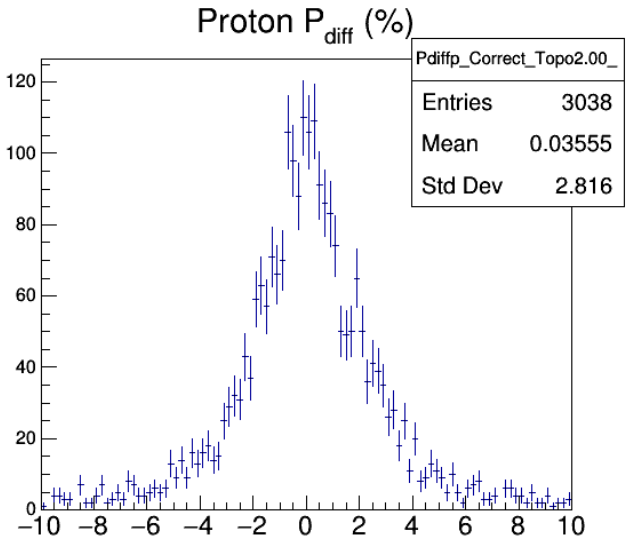


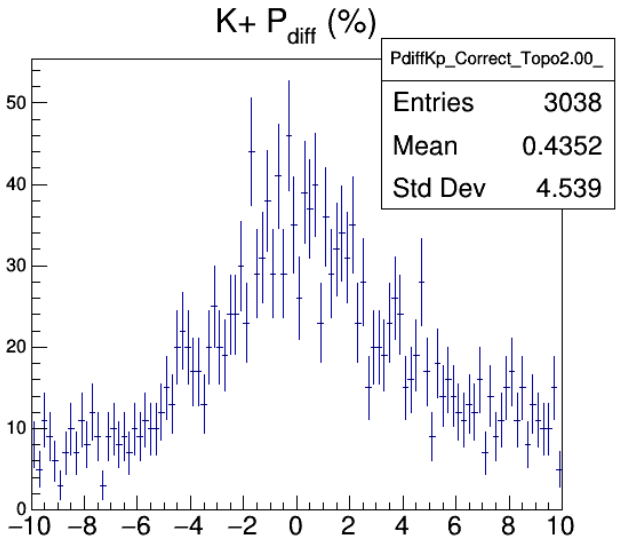


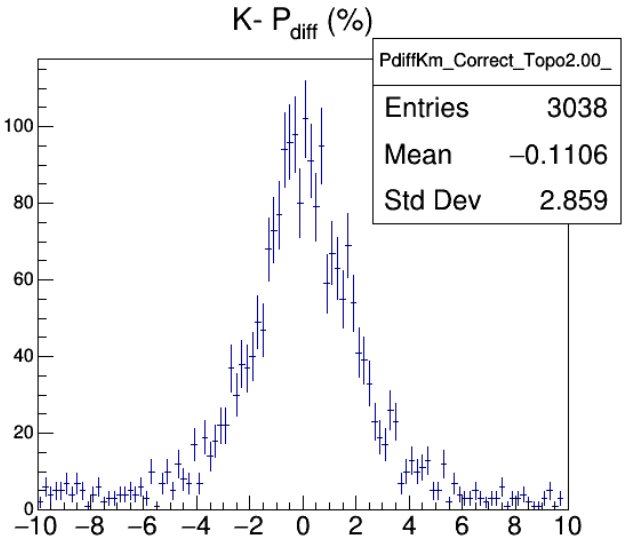


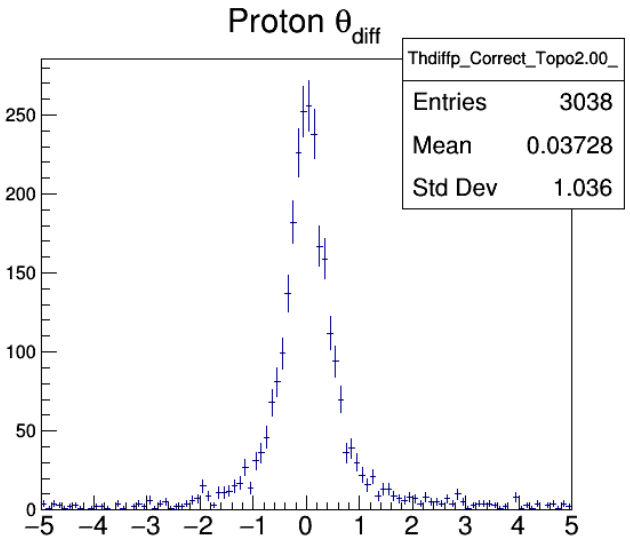
Detect pK+K-

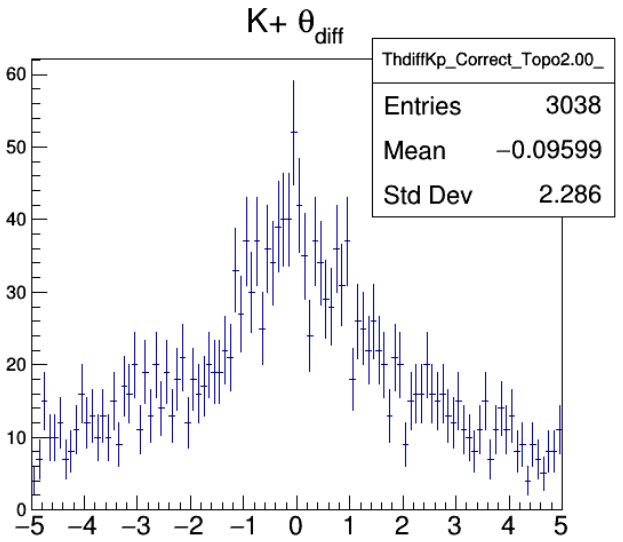
Detector Resolution Histograms

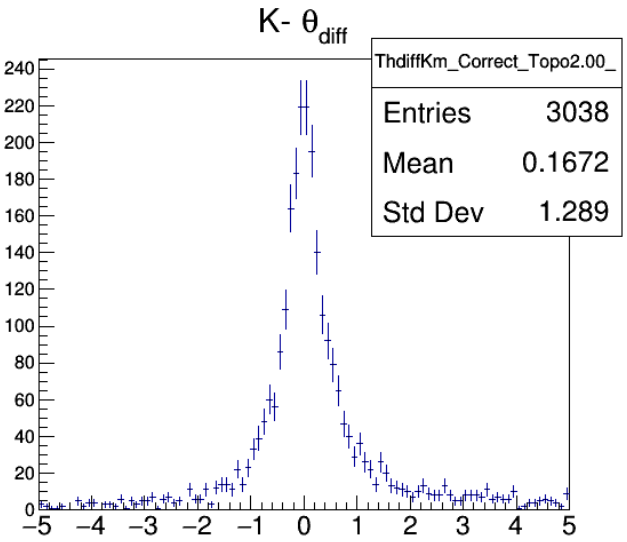












Detect pK-

Observable Extraction

- General Case N(>2) dimensional and we must account for all dimesions
- Need some combination of
- Bin over dimensions
- OR Integrate over dimensions
  - Difficult without model
- OR fit over dimensions
  - This is commonly done in Amplitude Analysis methods
  - Carlos Talk next + can extend to other problems…
  - Also see e.g. “Data analysis techniques, differential cross sections, and spin density matrix elements for the reaction $\gamma p \rightarrow \phi p$”, B.Dey+CLAS
- e.g Polarisation observables, spin density matrix elements, angular moments,…
- Also acceptance corrected yields are side product

Extended ML with acceptance

$$L(p) = \prod_i^N \frac{f(\tau_i : p)}{\int f(\tau_i : p)\, d\tau}$$

Standard likelihood, product Prob. function f with obs τ and pars p

$$L_{acc}(p) = \prod_i^N \frac{f(\tau_i : p)\,\eta(\tau_i)}{\int f(\tau_i : p)\,\eta(\tau_i)\, d\tau}$$

With acceptance take product of f and acceptance function

$$A(p) = \int f(\tau_i : p)\,\eta(\tau_i)\, d\tau$$

Probability Normalisation ∫

$$L^{ext}_{acc}(p) = \left[\frac{A(p)^N}{N!} e^{-A(p)}\right] \prod_i^N \frac{f(\tau_i : p)\,\eta(\tau_i)}{A(p)}$$

Extended ML with Poisson statistics

$$-\ln L^{ext}_{acc}(p) \propto -\sum_i^N \ln f(\tau_i : p)\,\eta(\tau_i) + A(p)$$

Simpler to minimise ln ignore terms independent of p

$$= -\sum_i^N \ln f(\tau_i : p) - \sum_i^N \ln \eta(\tau_i) + A(p)$$

$$\propto -\sum_i^N \ln f(\tau_i : p) + A(p)$$

Need to minimise this

$$A(p) \simeq \sum_j^M f(\tau_j : p)$$

Approximating A as sum of f over M accepted Monte-Carlo events

New RooHSAbsEventsPDF class

RooFit PDF classes require normalisation

- -This can be done by users own method
- -Or RooFit performs is own numerical integration(vegas)

RooHSAbsEventsPDF calculates its own integral by summing over MC events – includes partial integration for plotting

Requires ROOT tree of reconstructed MC events with fit variables

Additionally can generate events using true values if these are past through in the tree

- Evaluate function with gen, produce events with rec

Tests if integral is constant (can be ignored for speed)

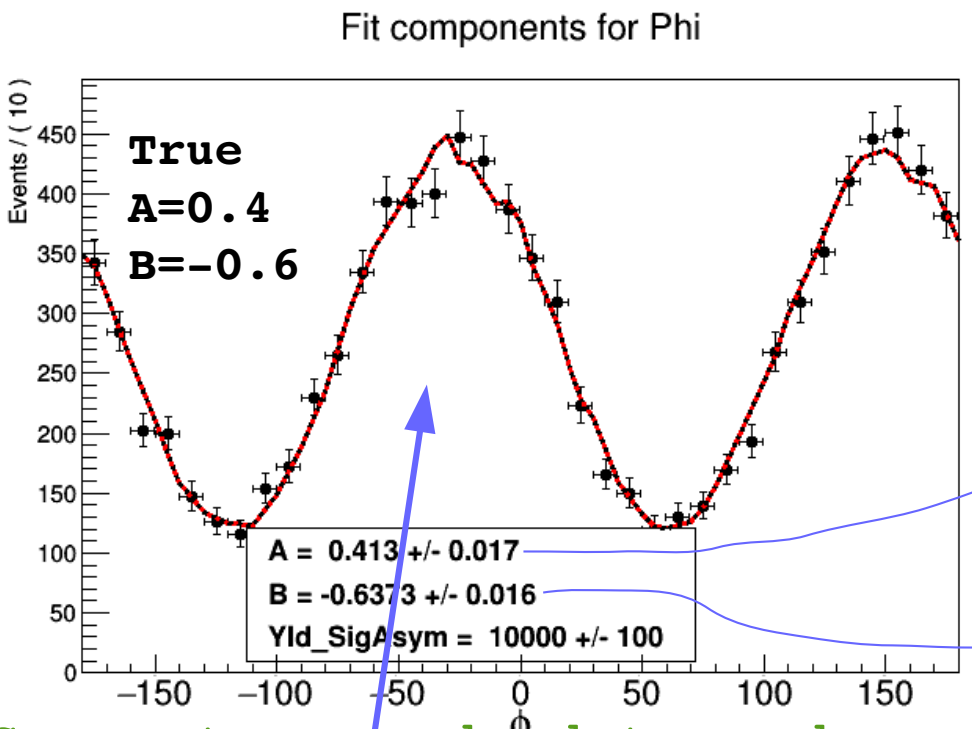
Works any number of fit observables or parameters

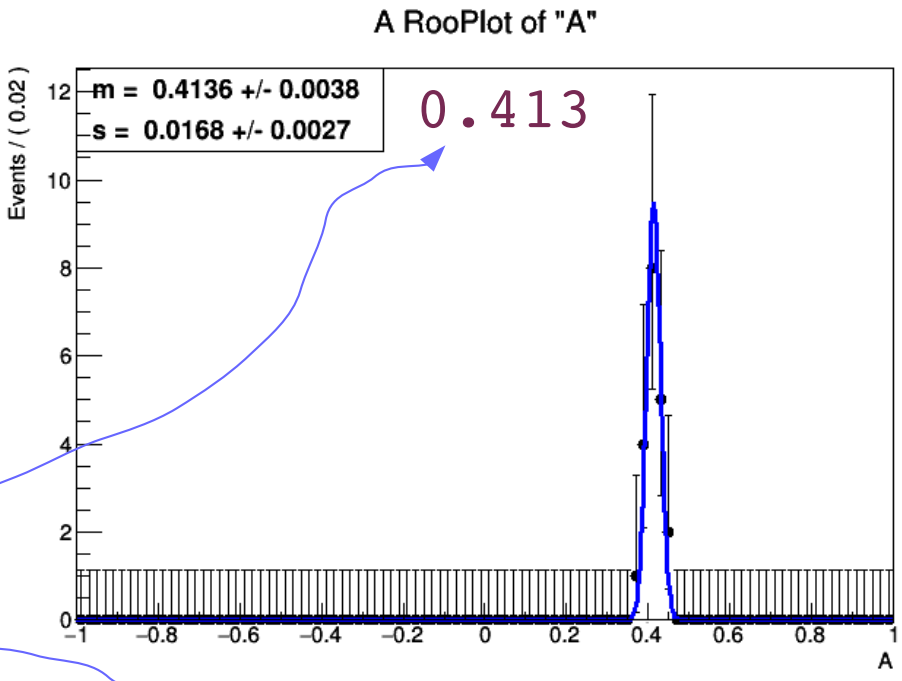
- - i.e. given a Ndim Model it will fit for Mpar parameters accounting for detector acceptance effects

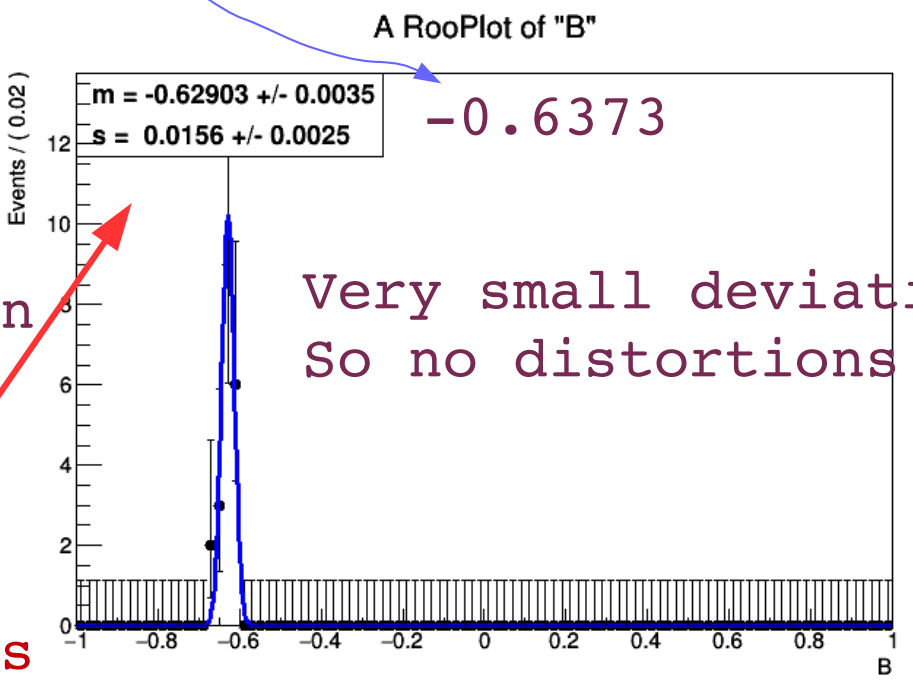
Observable Validation : RooFit MCStudy

- Extracted fit parameters will not necessarily be good estimates of true values
- Systematic errors from acceptance/resolution/bias/…
- Brute force method to estimate these is to perform many similar Toy fits using MC data
- Deviation from true values should correspond to deviations of fits to real data
- This is automated when using RooHSAbsEventsPDF and THSRooFit manager
- Just requires large amount of simulated data!
- Note: Generated(Truth) variables are used to generated reconstructed variables are used to fit
- Can use either accept/reject or weights for generation
- Can also be used to study planned reaction analysis
  - i.e. for CLAS12

Study Fit on ideal data







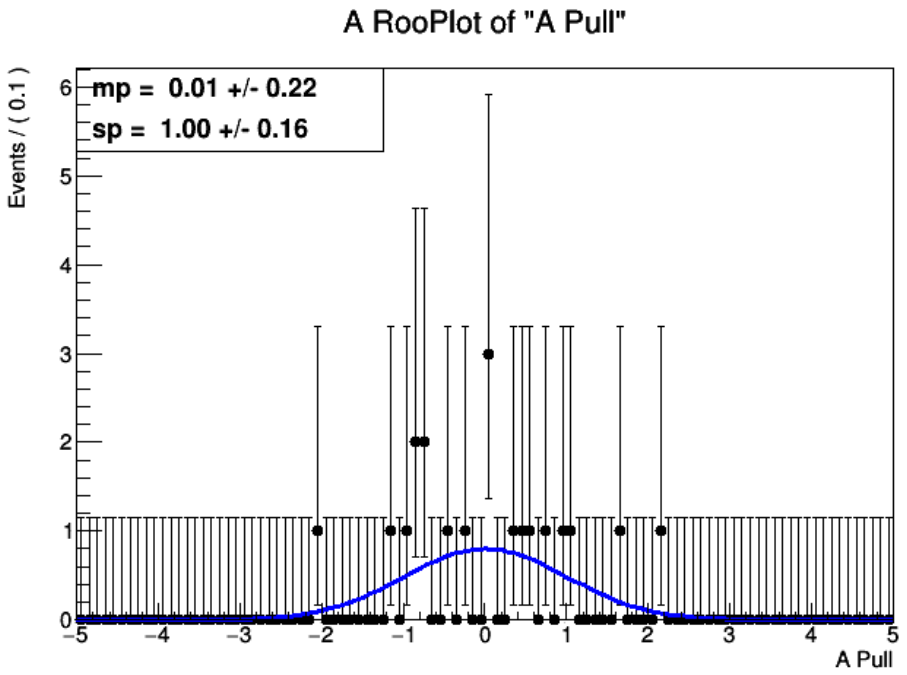
Generate pseudo data and simulation outwith RooFit :

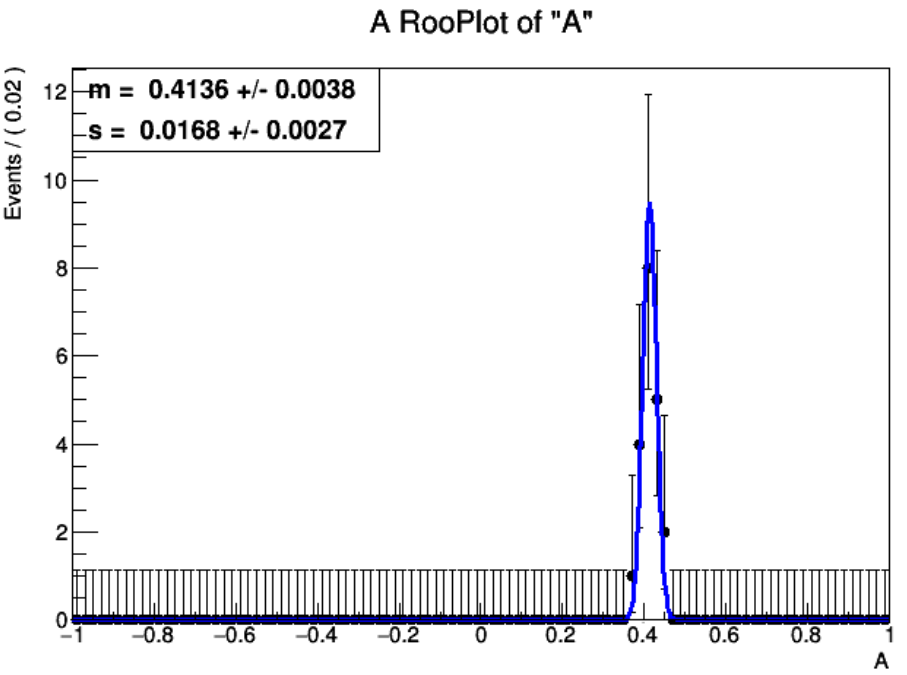
(1+A*P*cos(2Φ)+B*P*sin(2Φ))
P varies from 0.5-1 handled on event level

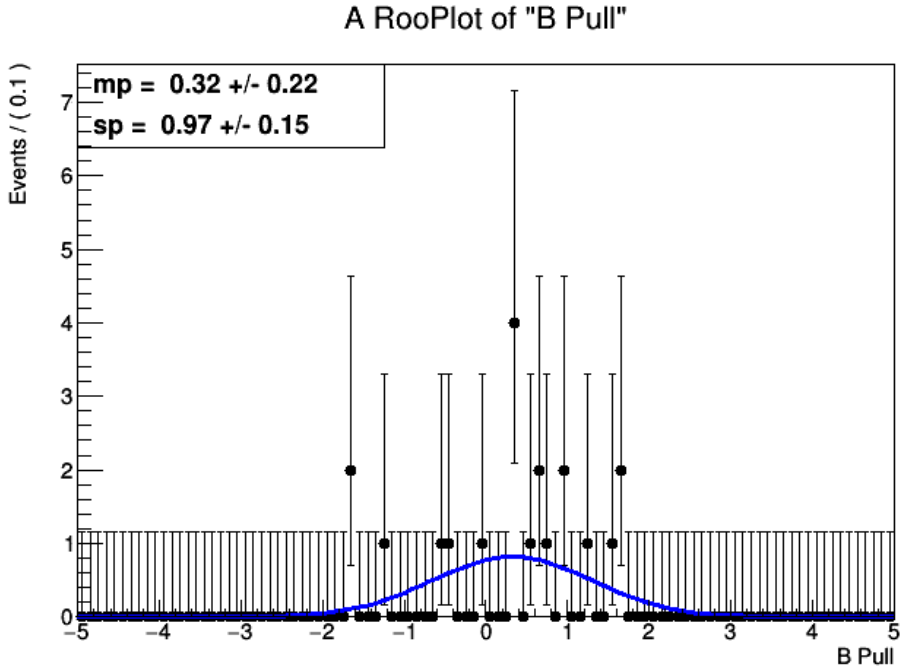
Normal fit to pseudo data
Study 20 fits to toy data based on normal fit parameters

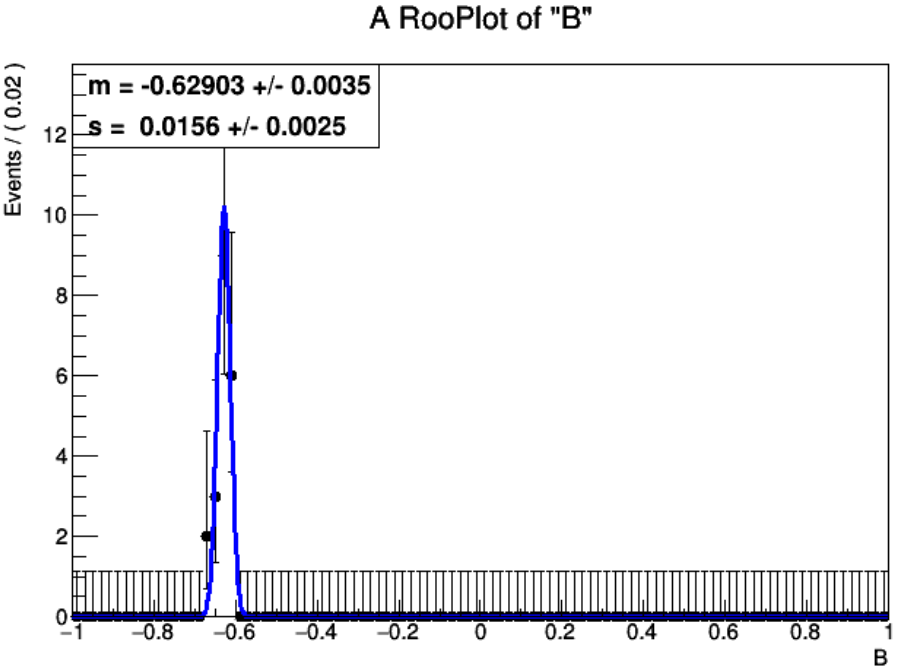
Very small deviations
So no distortions

Study Fit on ideal data



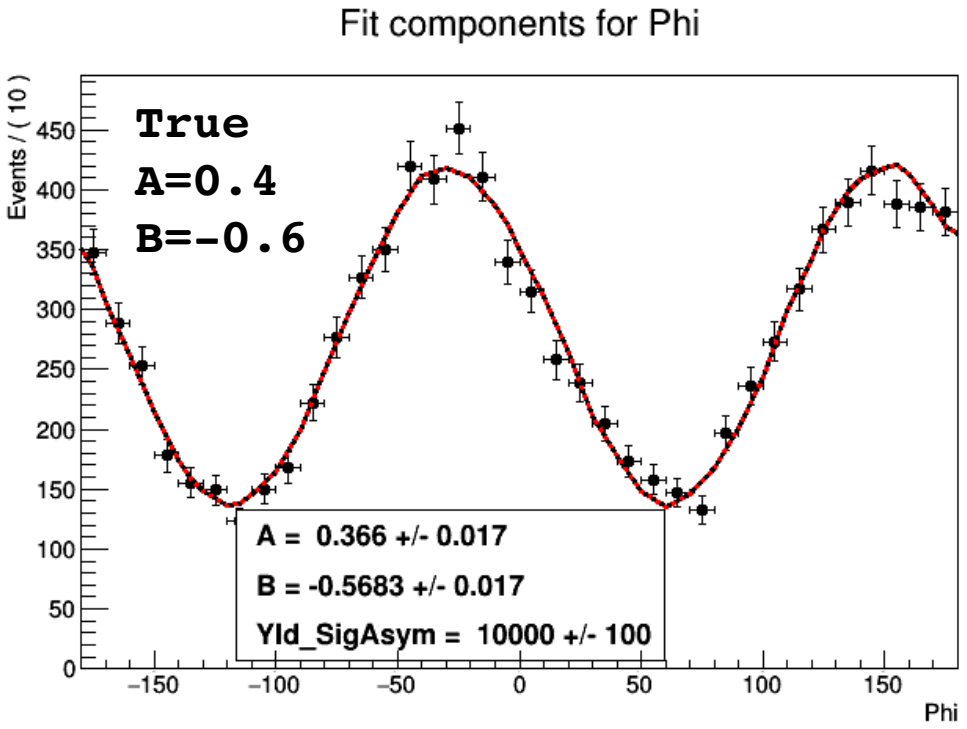


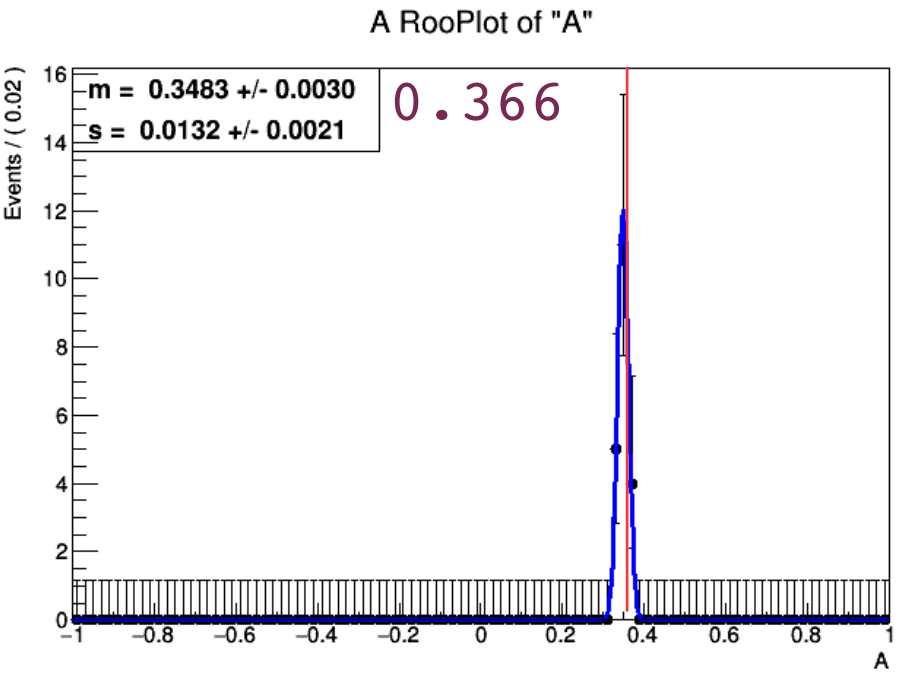


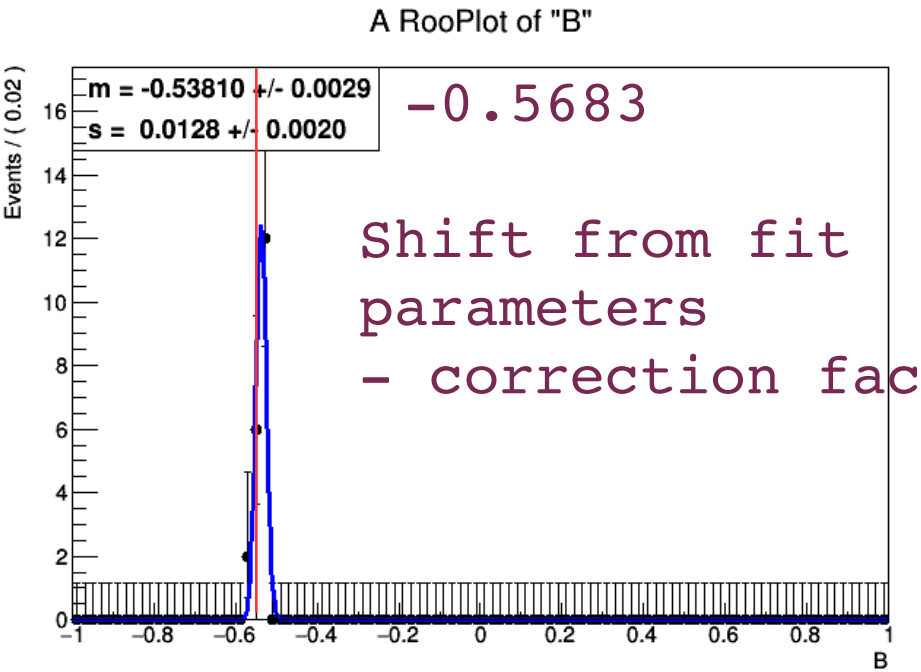


Study Fit with resolution

Add 10° resolution to Φ
-Causes slight dilution





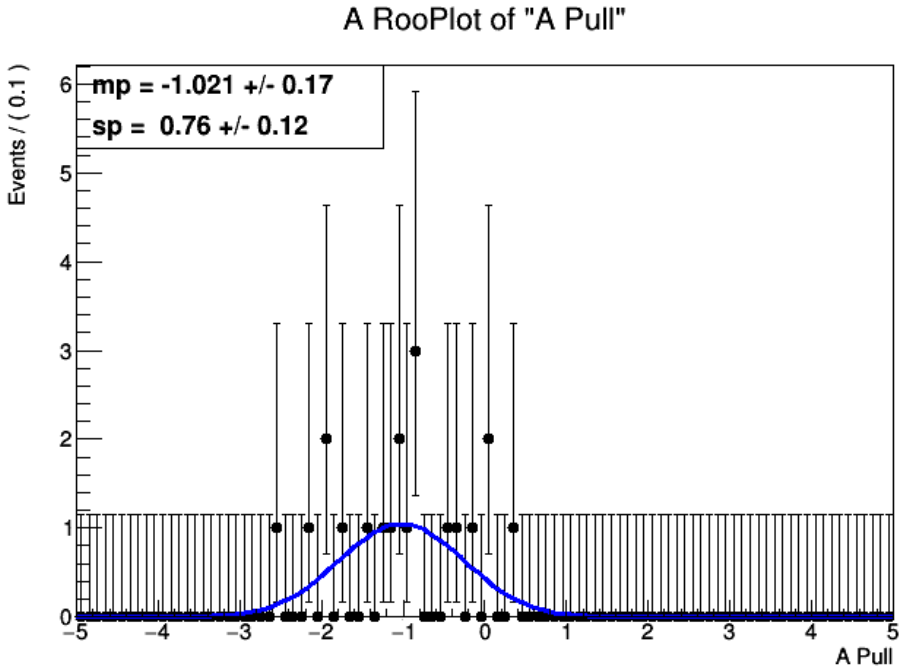


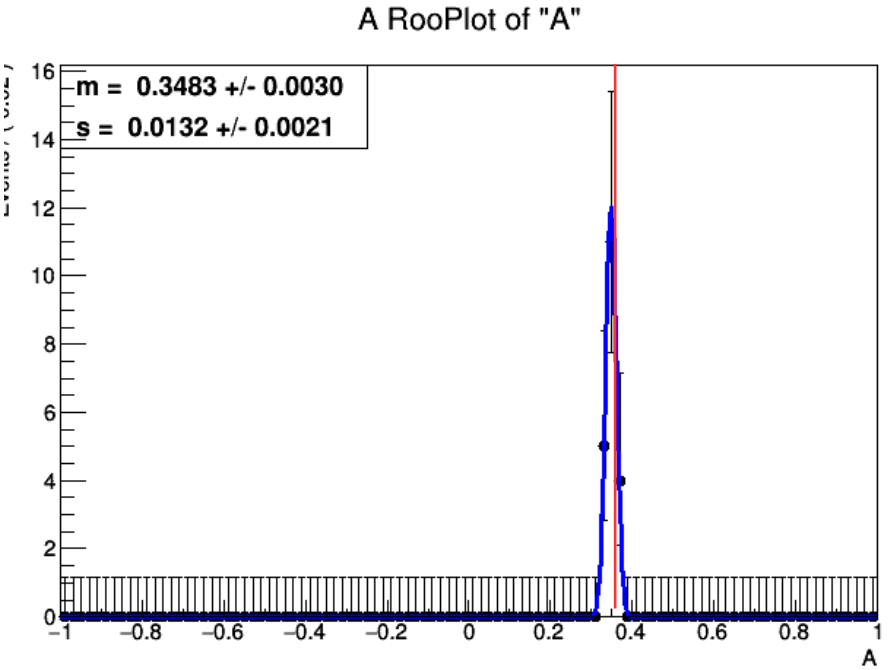
Shift from fit parameters
- correction factor

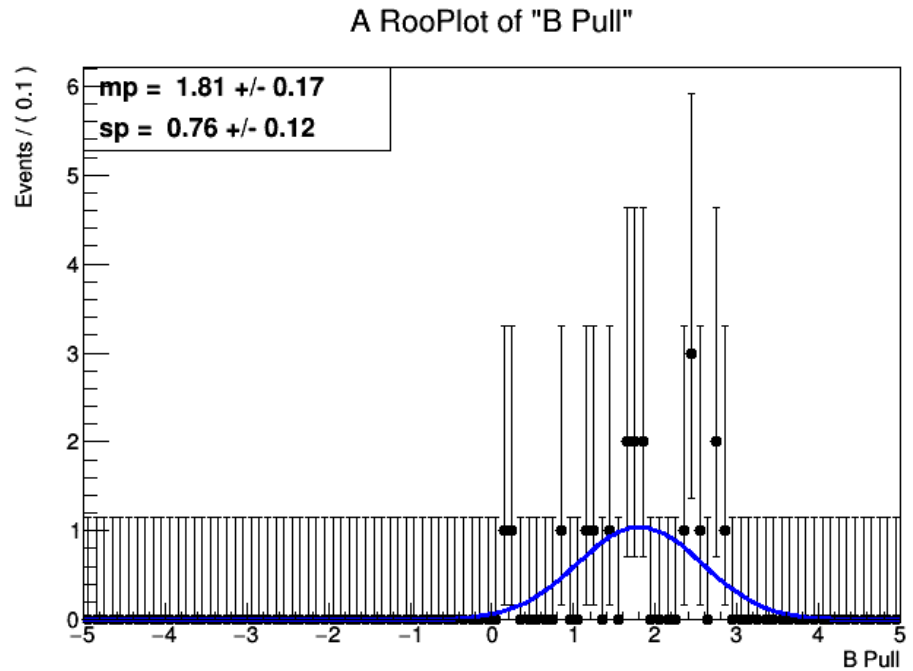
With Corrections applied:

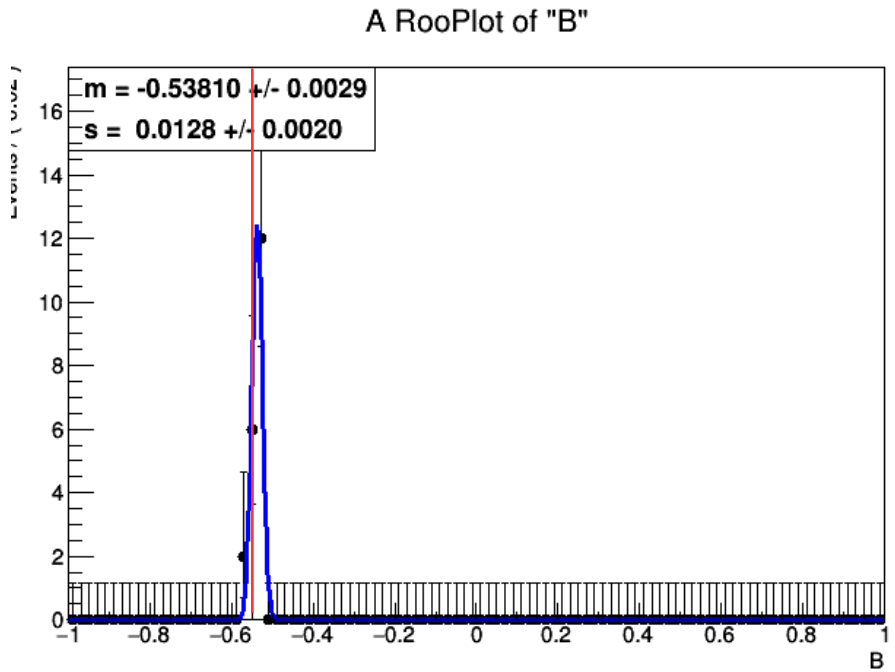
A = 0.383724 +/- 0.0173657
B = -0.598552 +/- 0.0165525

Study Fit res 10 no acceptance

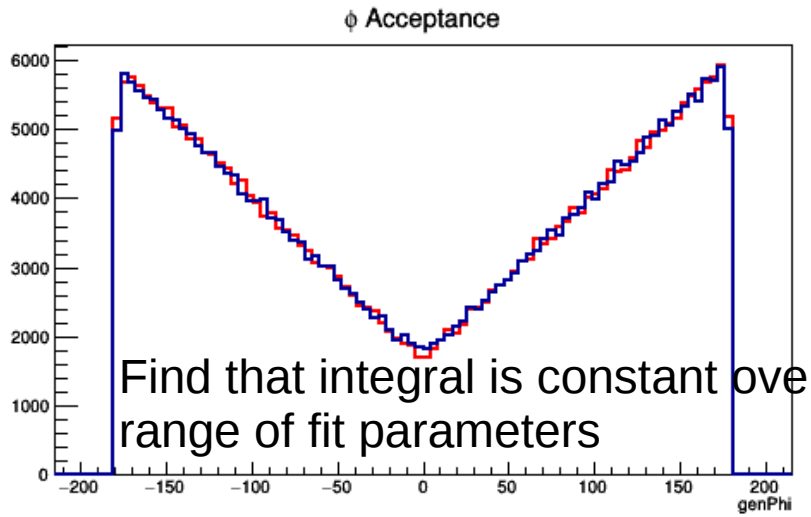




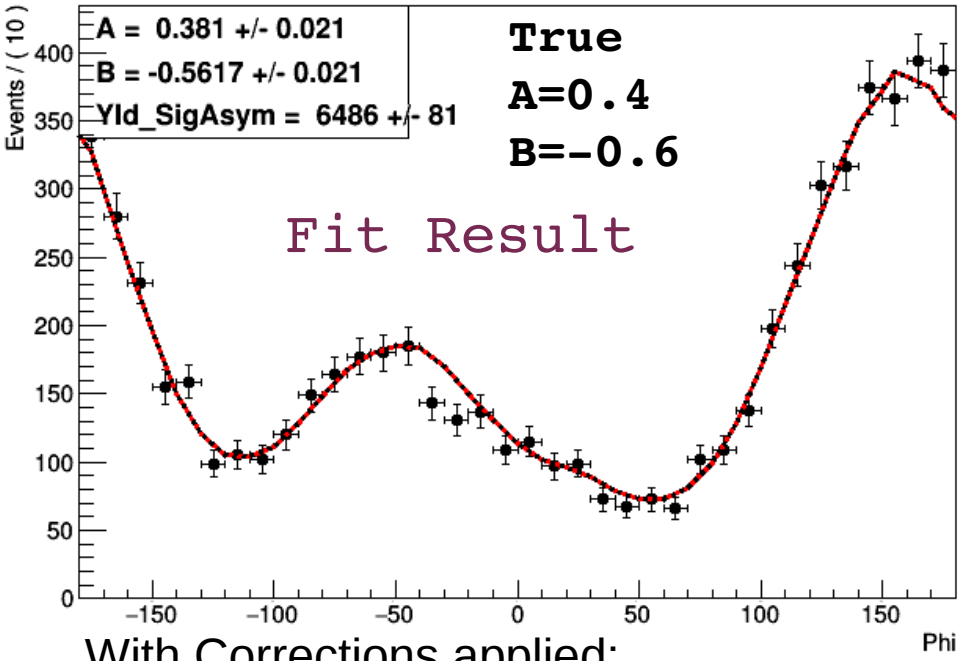




Add Symmetric acceptance + Res



Find that integral is constant over range of fit parameters

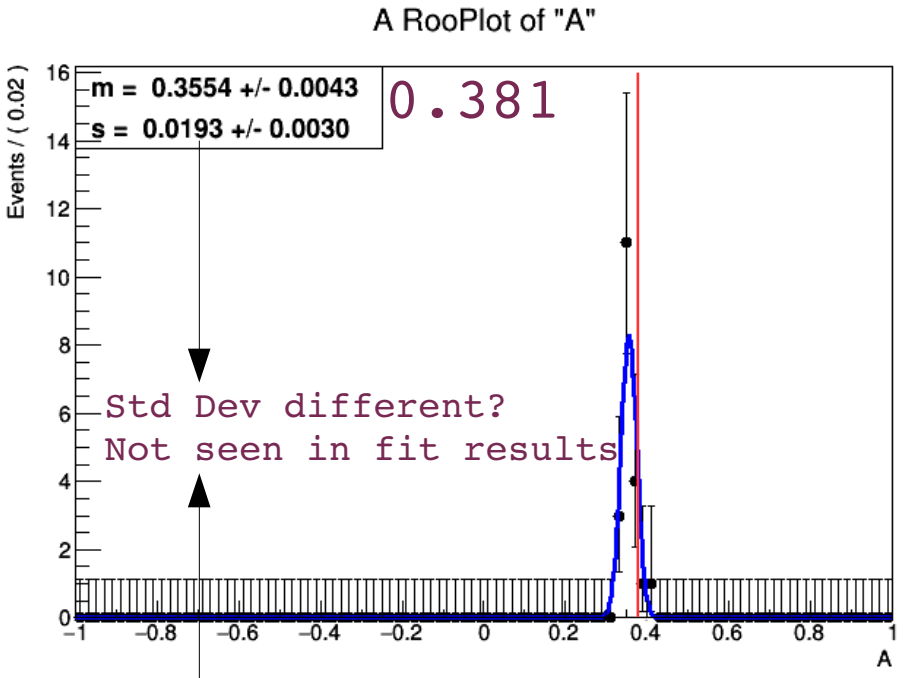


True
A=0.4
B=-0.6

Fit Result

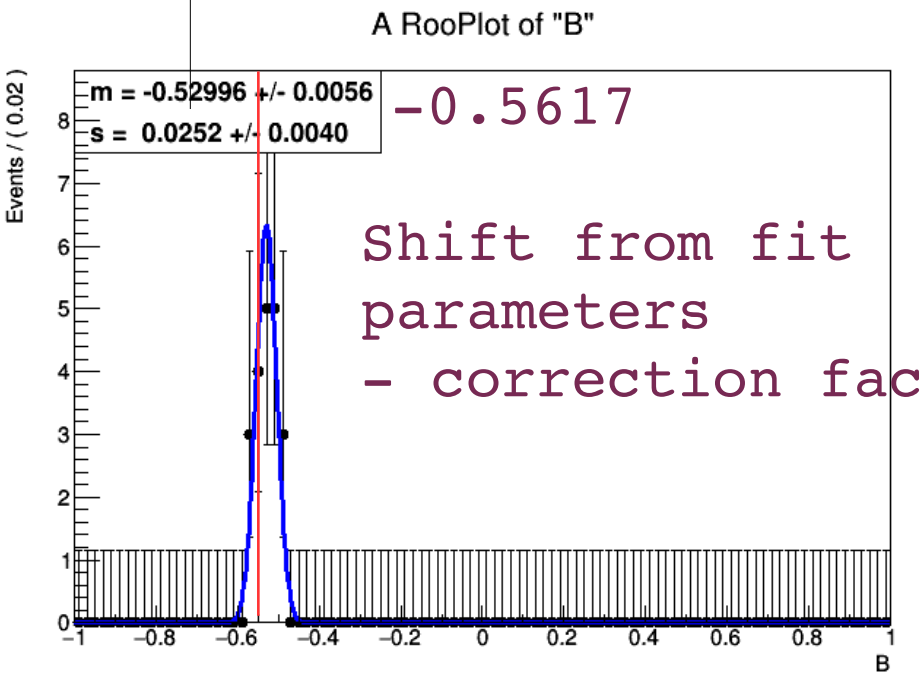
With Corrections applied:

A = 0.405625 +/- 0.02131
B = -0.593406 +/- 0.0207043



0.381

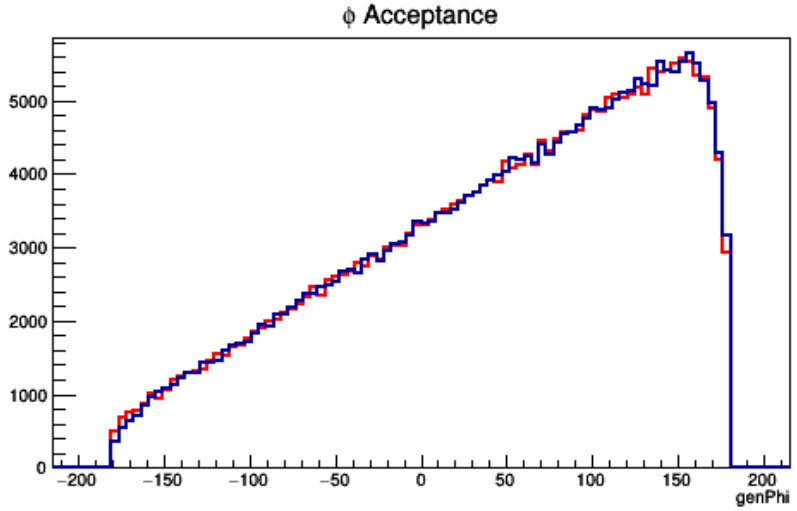
Std Dev different?
Not seen in fit results

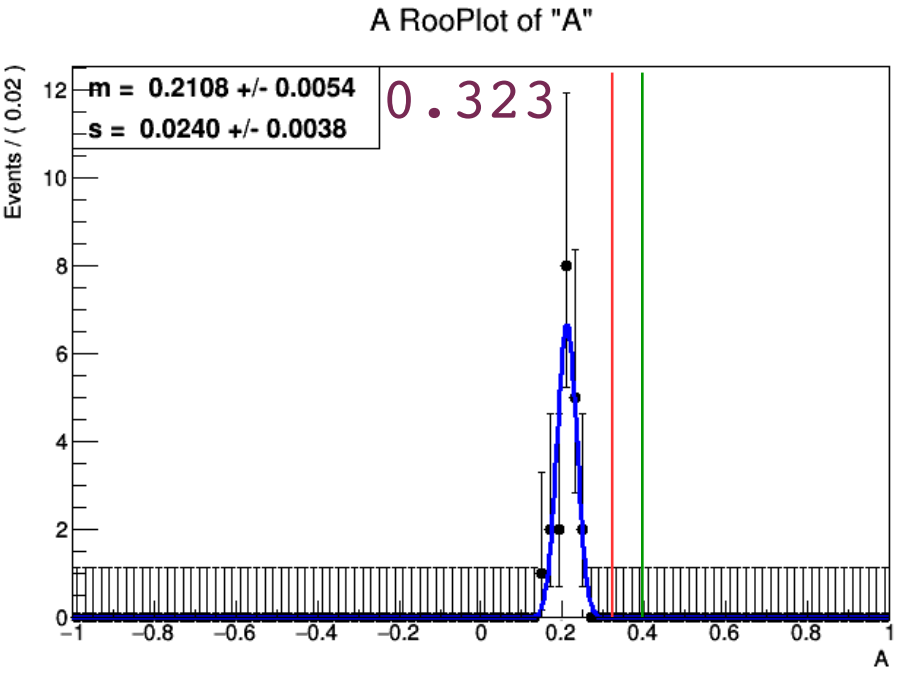


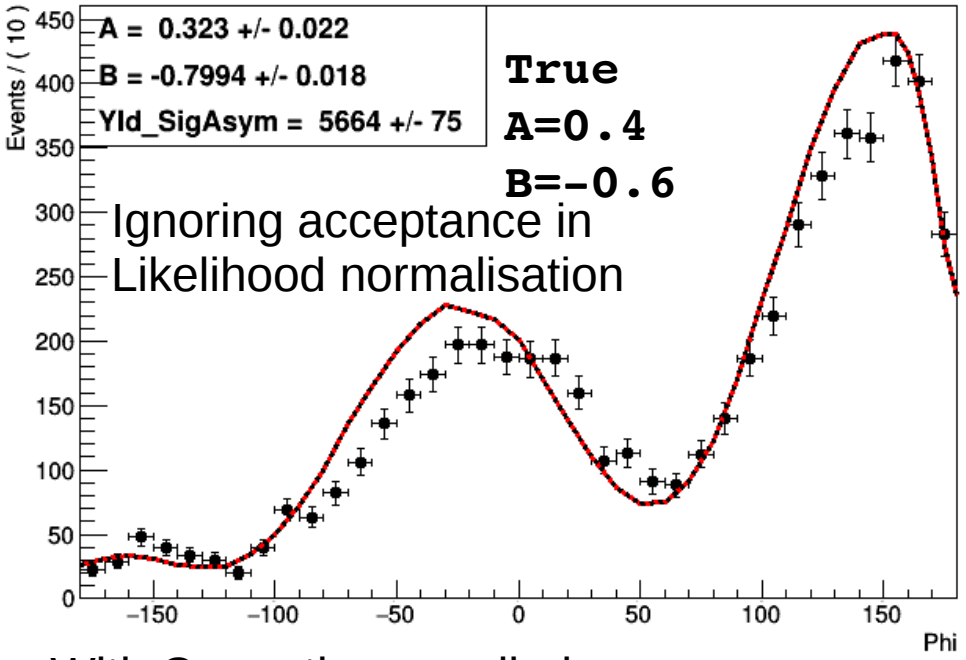
-0.5617

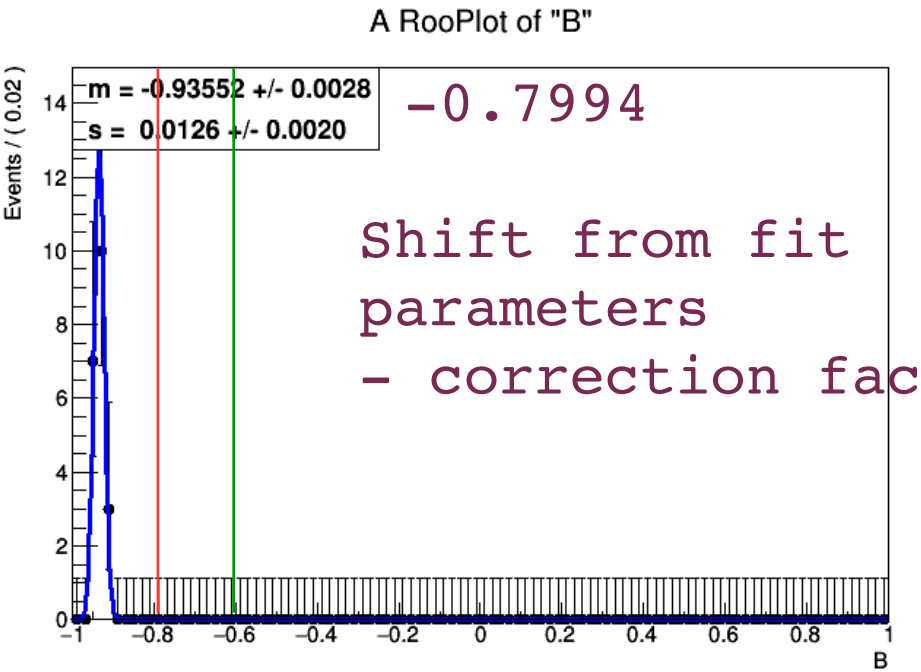
Shift from fit parameters
- correction factor

Add Asymmetric acceptance + Res



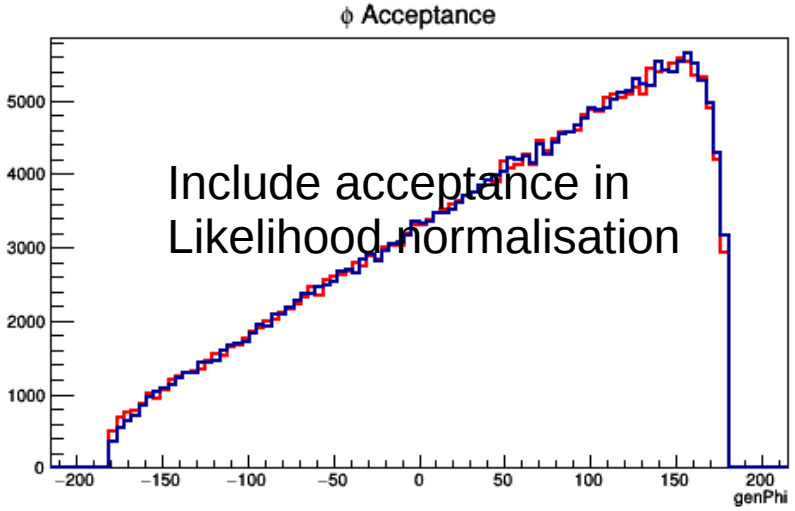


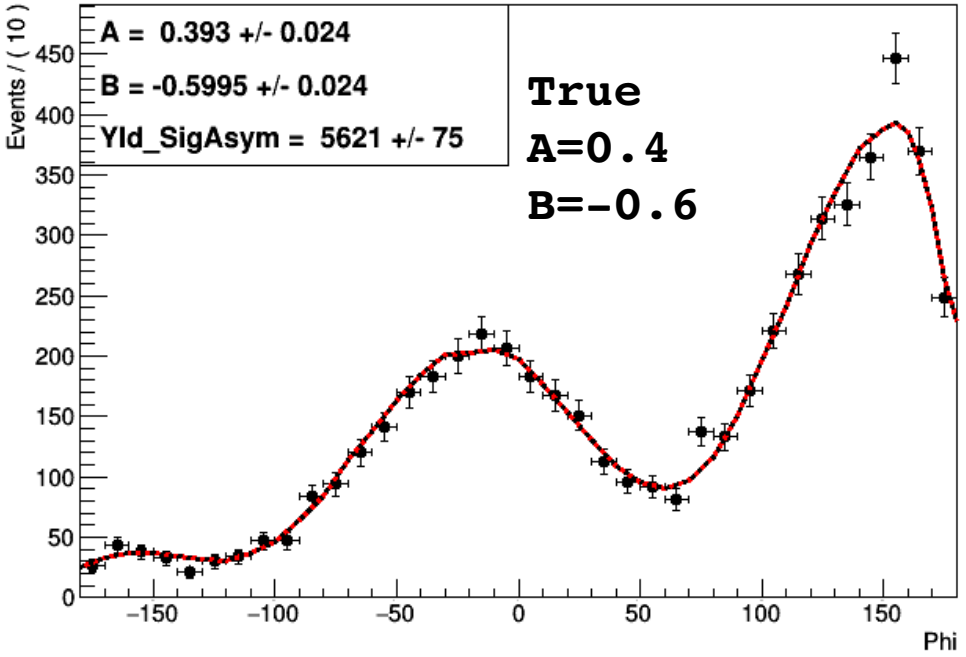




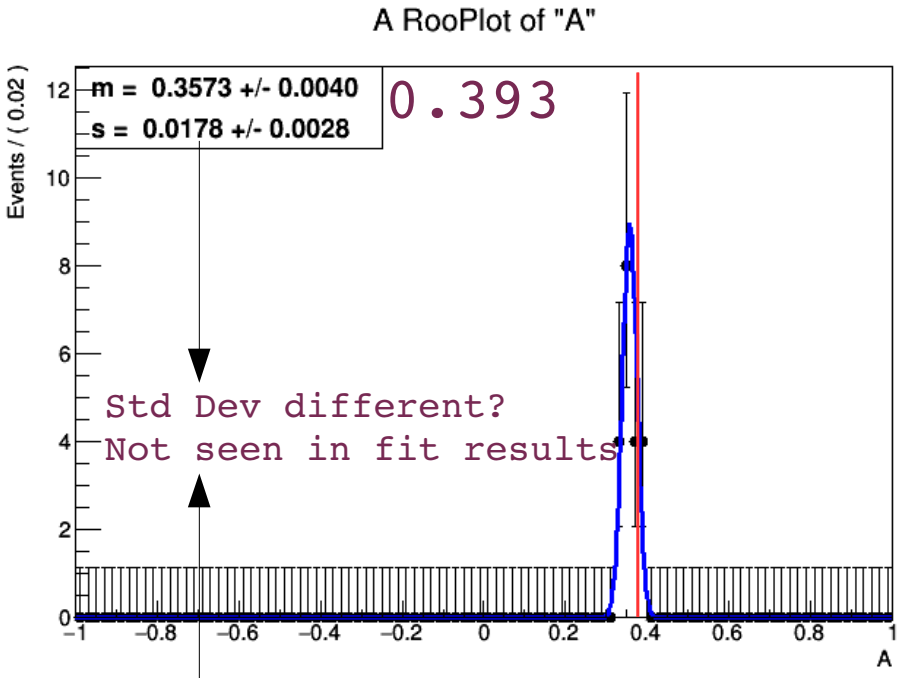
With Corrections applied:
A = 0.436134 +/- 0.0224558
B = -0.663225 +/- 0.017903

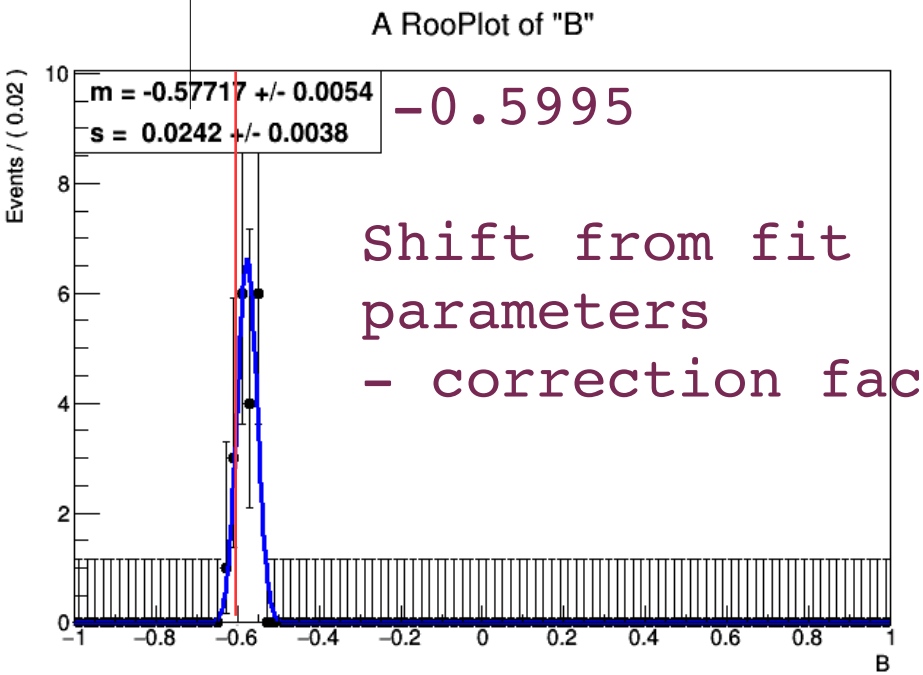
Add Asymmetric acceptance + Res



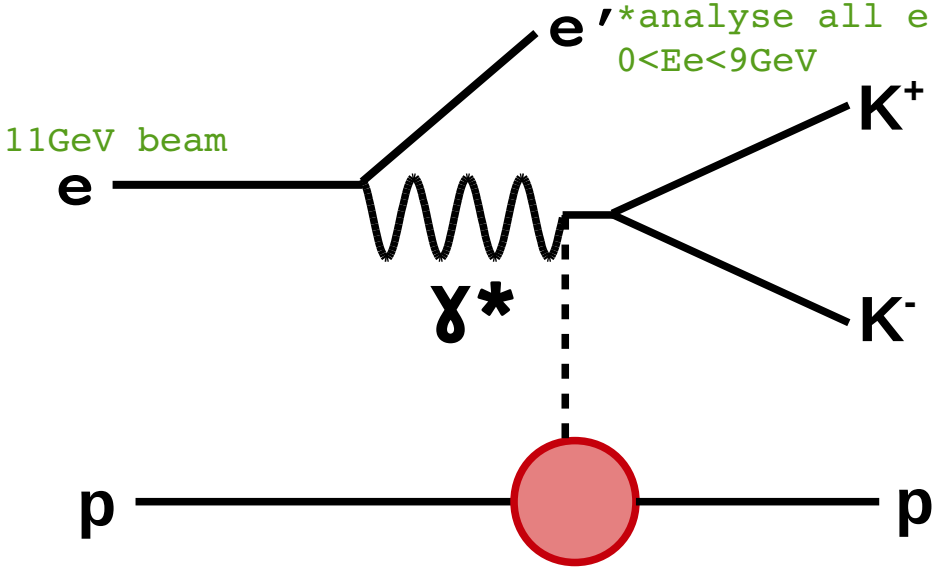


With Corrections applied:
A = 0.428165 +/- 0.0236552
B = -0.621748 +/- 0.0239505





Toy Example Full Chain Analysis



Simulate 40k phase space events with gemc

Reconstruct with coatjava

Write to text with groovy

Convert to ROOT HSParticle

Calculate Fit variables with THSProject

Fold in model and fit with hsfit

Production mechanism :

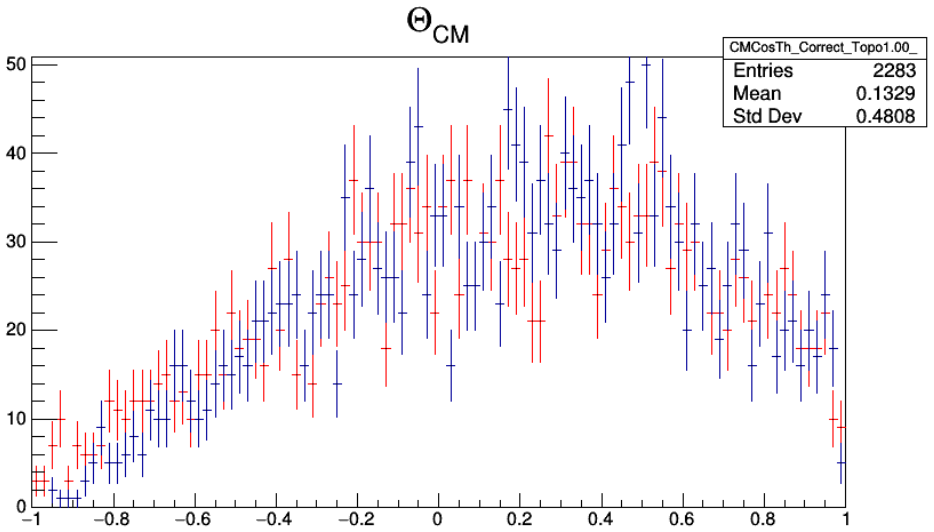
$[1 + A*P(\gamma)*\cos(2\Phi)][1+B*\cos(\Theta)]$

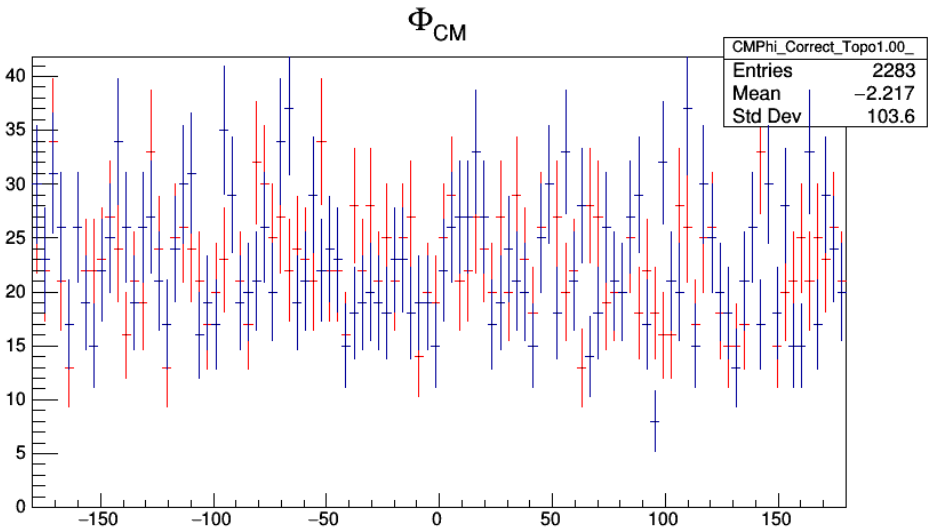
Decay Mechanism :

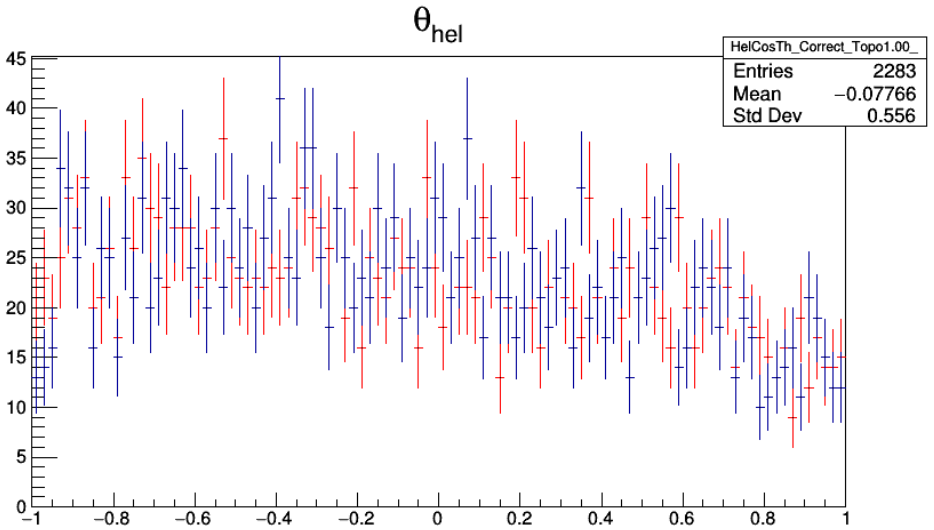
$[1+C*\cos(2\varphi)+ D*\sin(2\varphi)][1+ E*P(\gamma)*\cos(\theta)]$

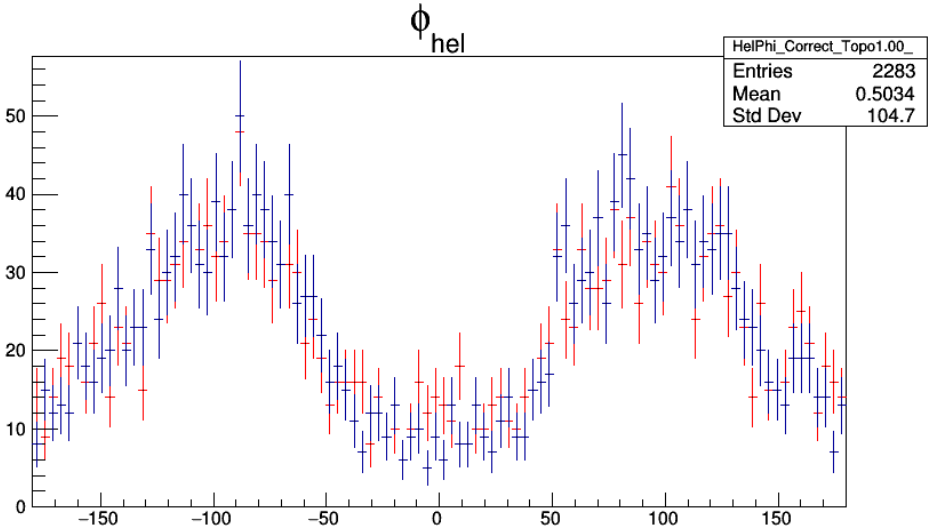
Need to extract 5 parameters on 5 variables (inc. $P(\gamma)$)

Phase Space Angular Distributions eK+K-p



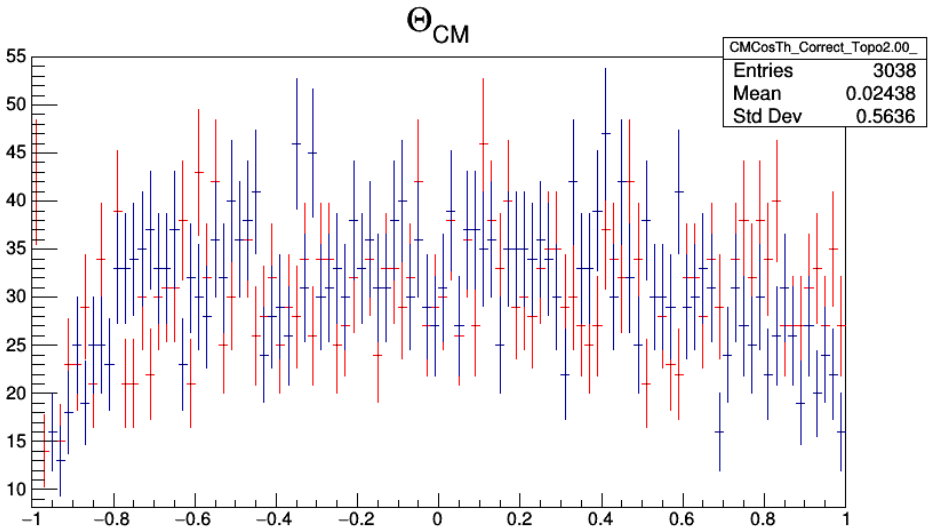


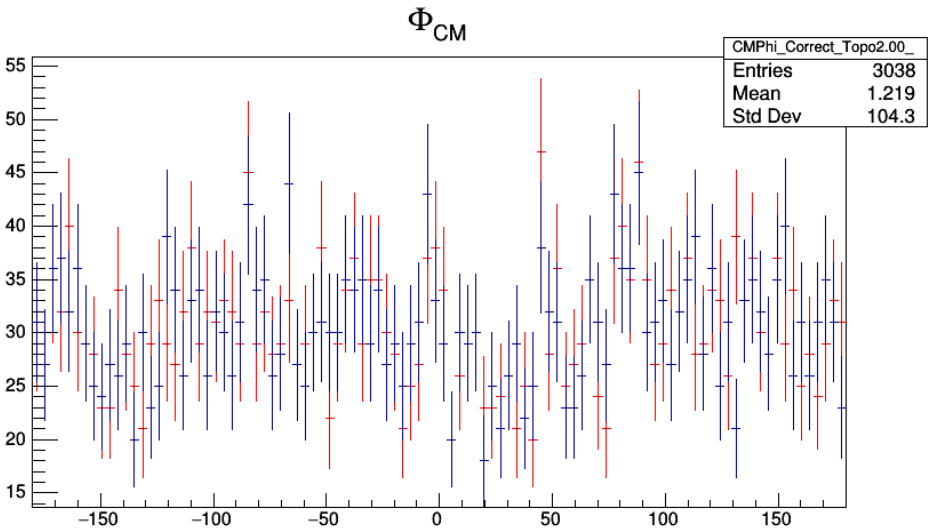


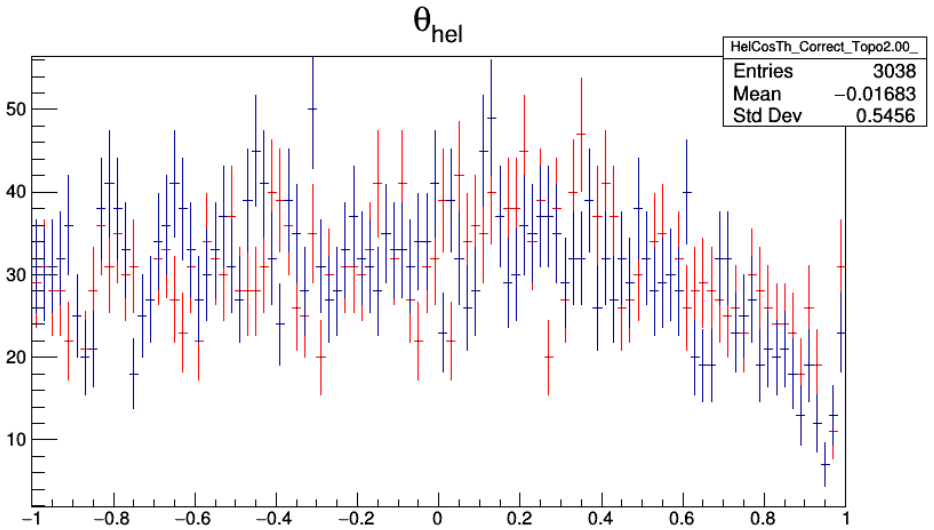


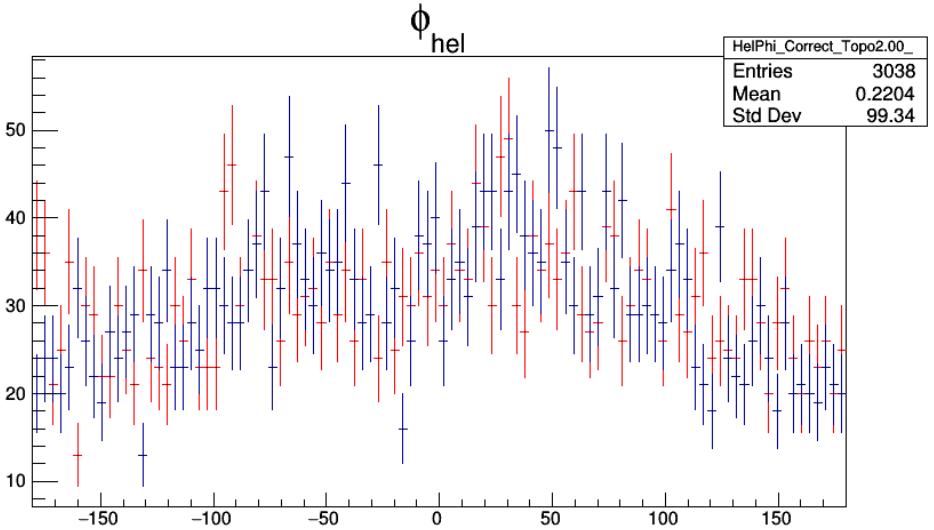
Reconstructed Generated

Phase Space Angular Distributions eK+p



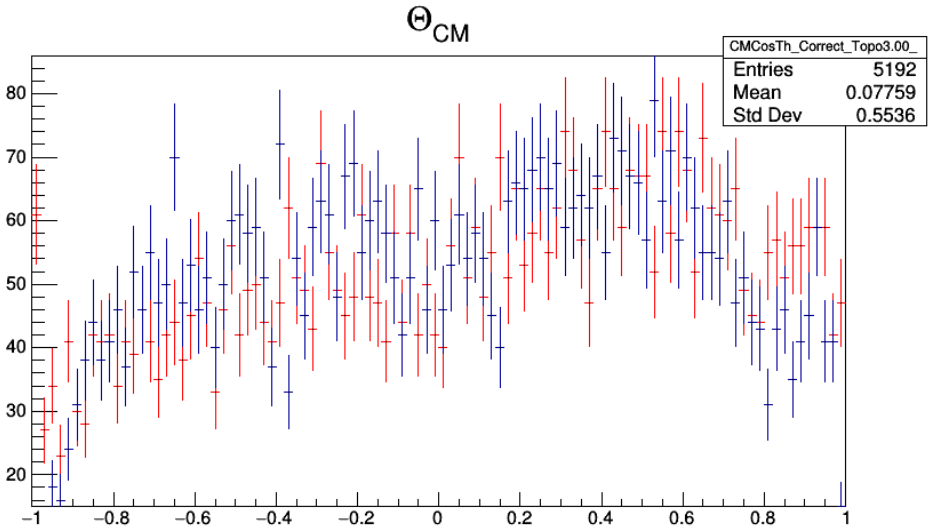


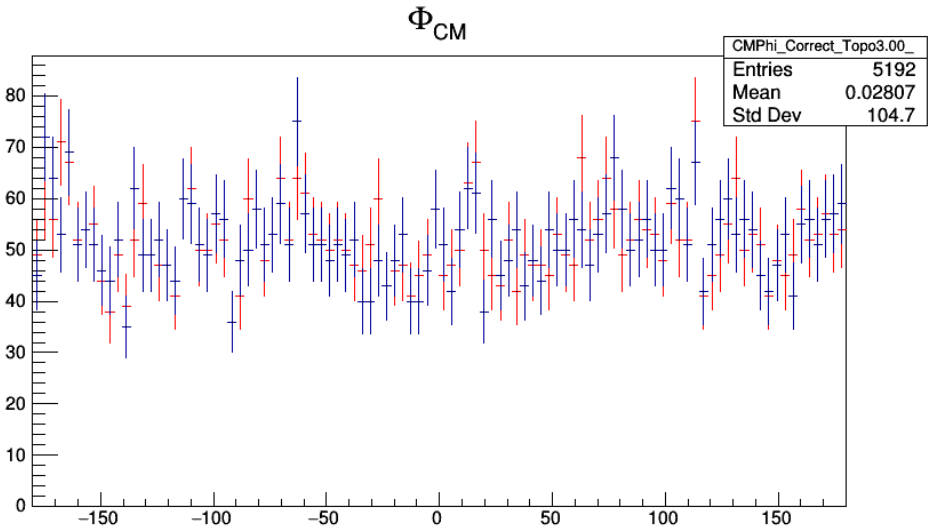




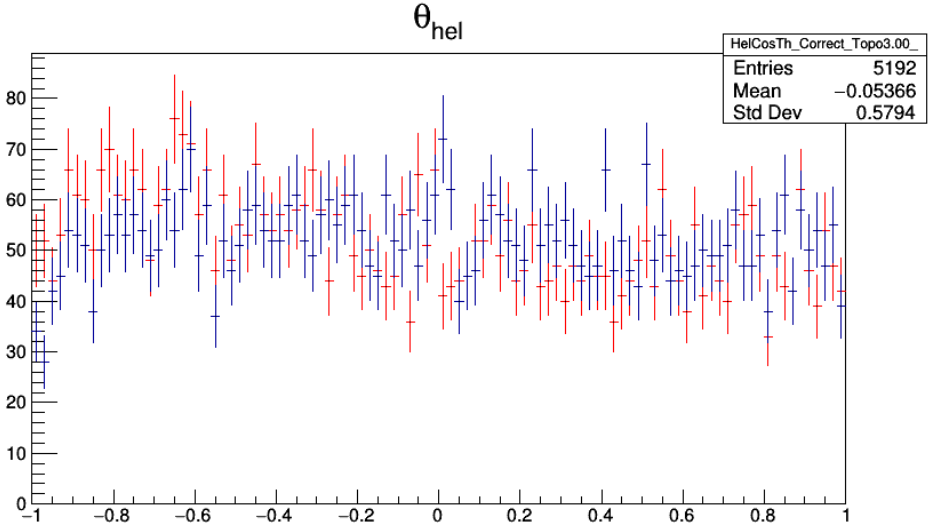
Reconstructed Generated

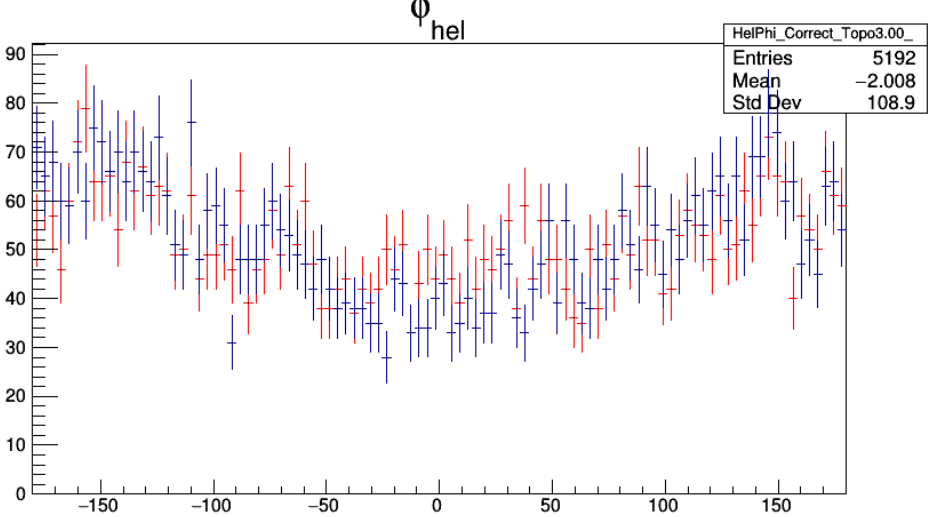
Phase Space Angular Distributions eK-p





Reconstructed Generated

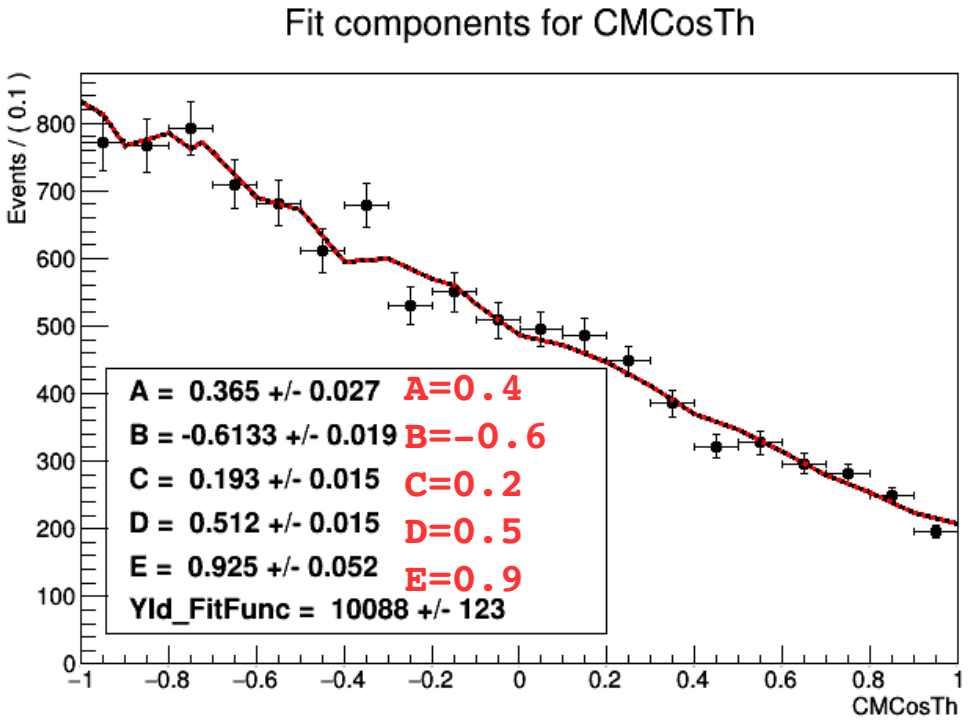


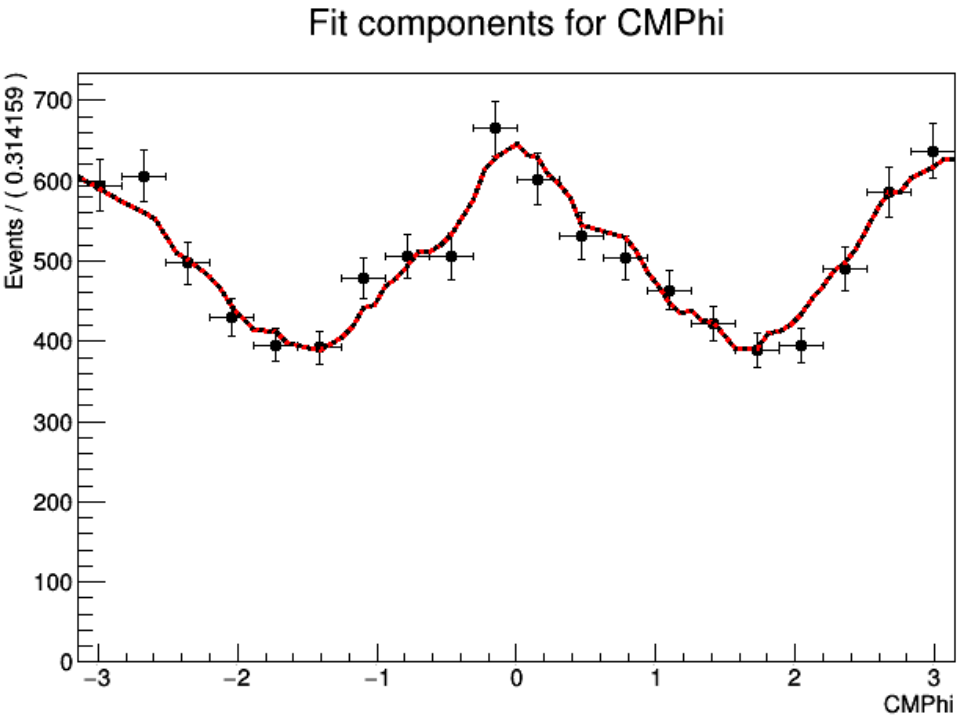


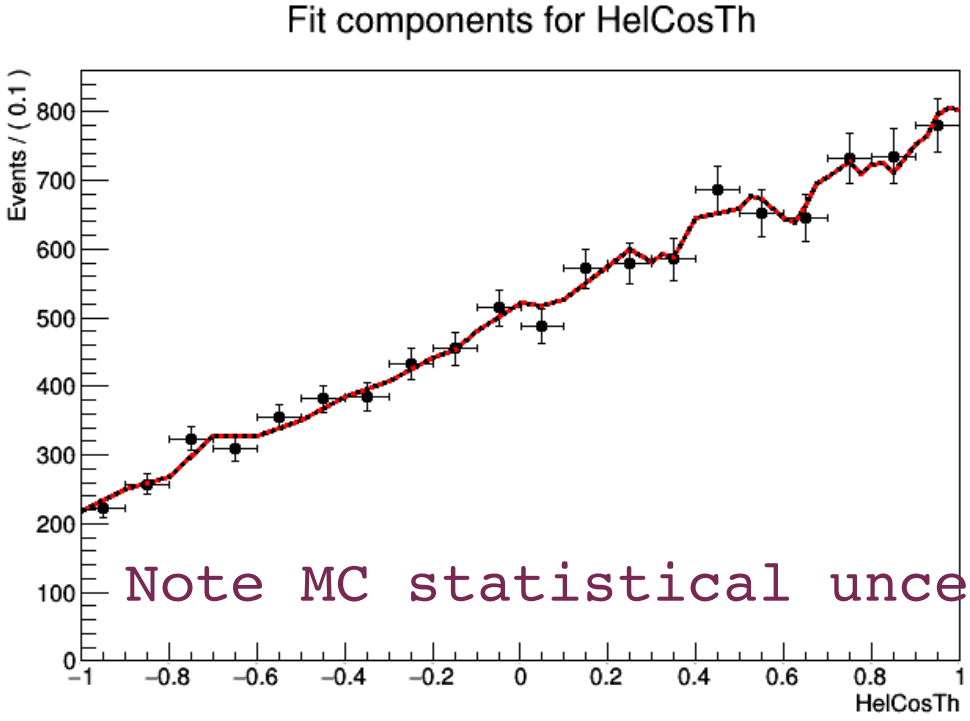
Will now use this data in toy model fit

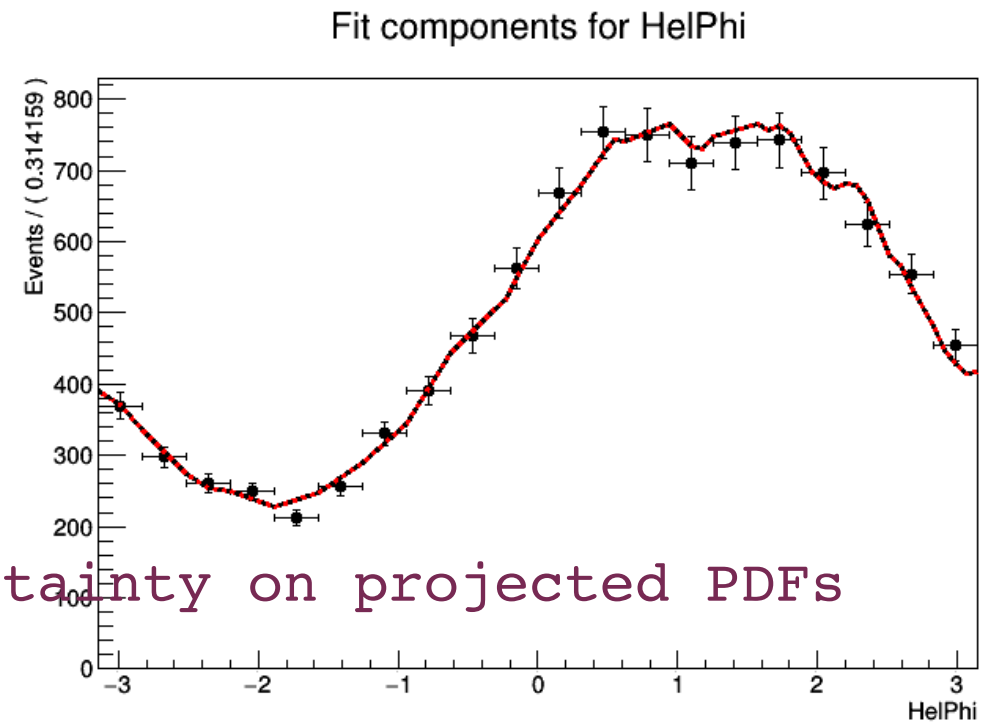
- Use red to generate blue to fit

Weight and fit GENERATED events



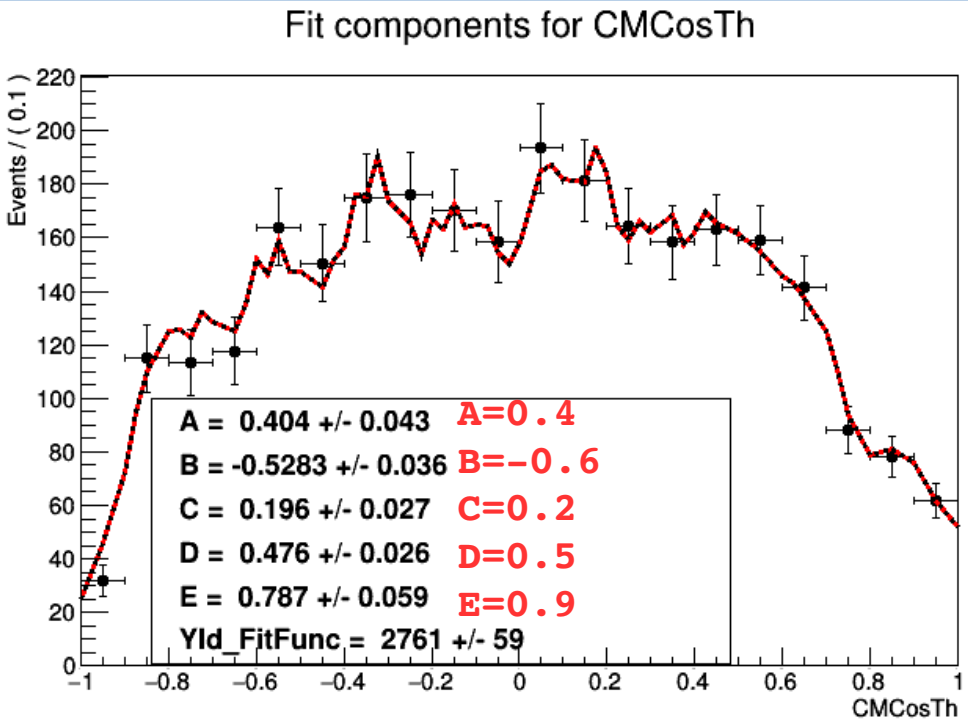


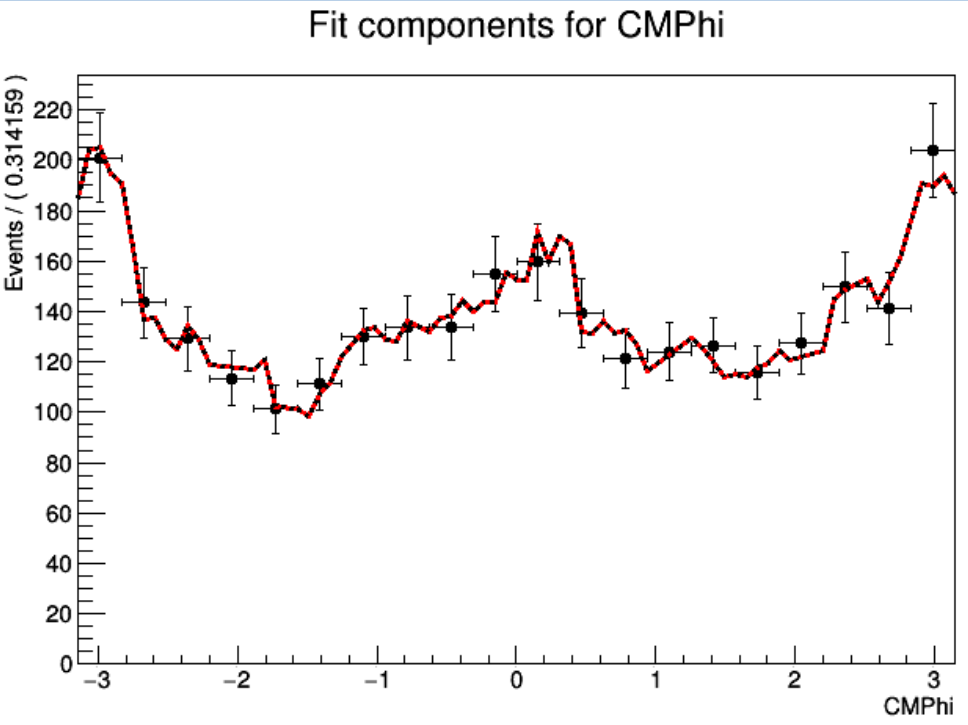


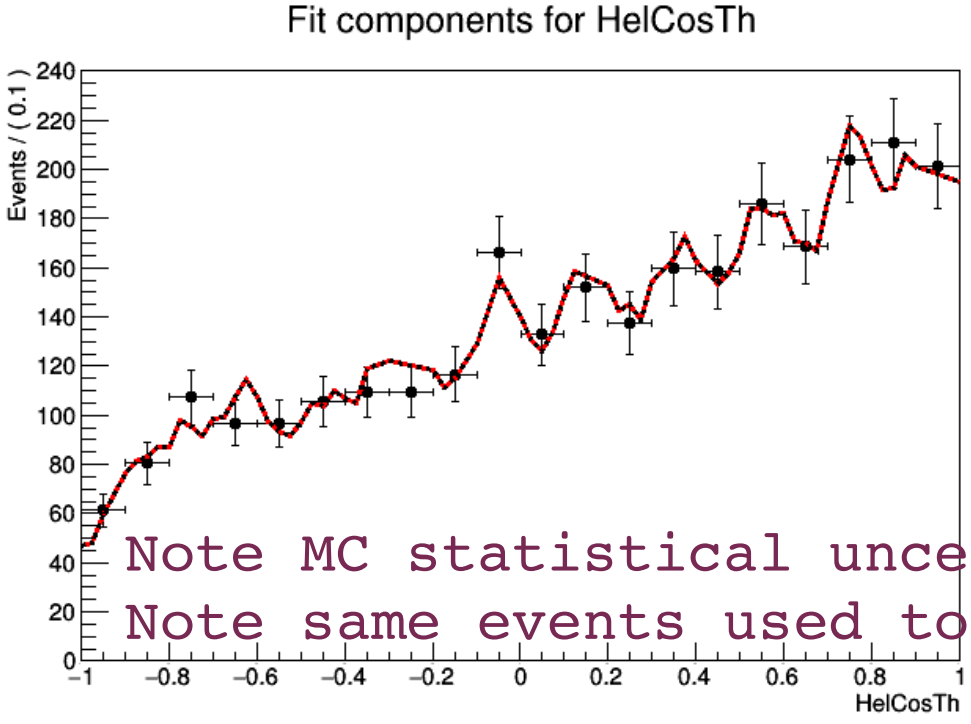


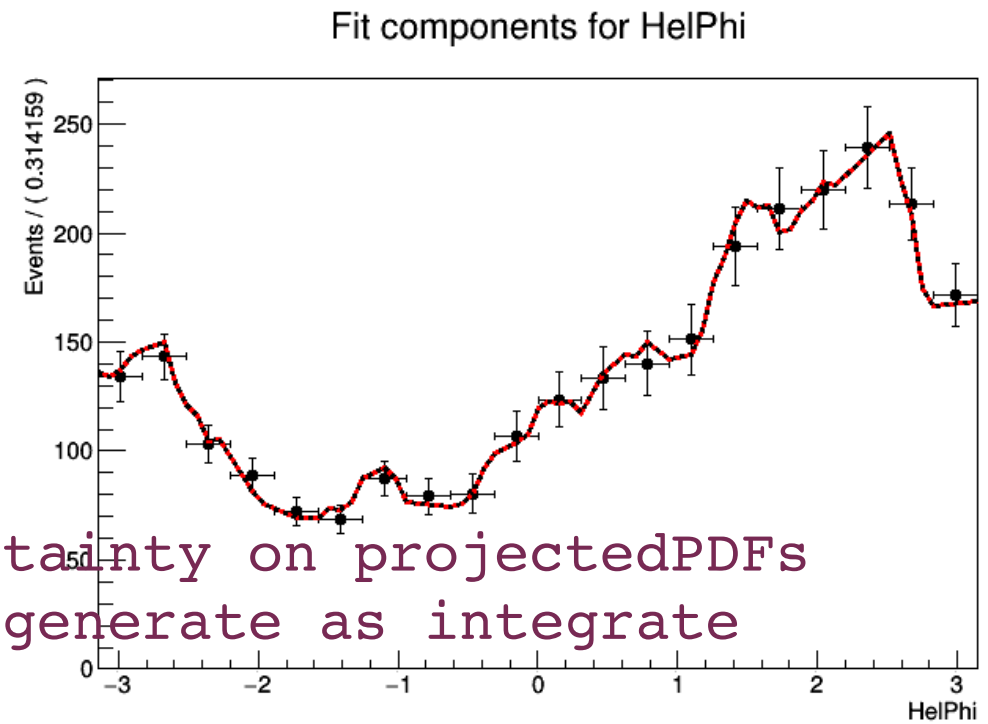
Note MC statistical uncertainty on projected PDFs

Weight and fit RECONSTRUCTED events









Note MC statistical uncertainty on projectedPDFs

Note same events used to generate as integrate

How to make a PDF?

Load Skeleton codemaker in ROOT

```
root --THSSkeleton.C
root [1] THSSkeleton skel;
Root [2] skel.CreateRooFitEventsPDF
("NewPDF","CMCosTh,CMPhi,HelCosTh,HelPhi,Pol","A,B,C,D,E")
```

ClassName Fit variables/observables Fit Parameters

Then edit file NewPDF.cxx to define fit function :

```
Double_t NewPDF::evaluate() const
 {
    return (1.0 + Pol*(A*TMath::Cos(2*CMPhi)))
     *(1+(C*TMath::Cos(HelPhi)+D*TMath::Sin(HelPhi)))
     *(1+B*CMCosTh)
     *(1+Pol*E*HelCosTh);
 }
```

Similar evaluateMC()

Run Script : Cuts and Bins

```
//root --hsfit ConfigureBins.C
{
  THSRooFit* RF=new THSRooFit("Binner");
  RF->SetOutDir("studyRecBins/");

  RF->LoadVariable("CMPhi[-3.14159,3.14159]");
  RF->LoadVariable("HelPhi[-3.14159,3.14159]");
  ...
  RF->LoadAuxVars("genCMPhi[-3.14159,3.14159]");
  ...
  RF->LoadBinVars("Correct",1,0.8,1.2);
  RF→LoadBinVars("Topo",3,0.3,3.5);
  RF->LoadBinVars("MissMass",1,0,1);

  TChain *chainRec=new TChain("FinalTree");
  chainRec->Add("../project/IncGamperm_eg_eKK_40k.root");
  RF->MakeBinnedTrees(chainRec,"FitFunc");

}
```

Run Script : MC Study

```
//root --hsfit Study.C --NewPDF.cxx
{
  THSRooFit* RF=new THSRooFit("AFit");
  RF->SetOutDir("studyRec4k/");

  RF->LoadVariable("CMPhi[-3.14159,3.14159]");
  ...

  RF->Factory("NewPDF::FitFunc(CMCosTh,CMPhi,HelCosTh,HelPhi,Pol,
      A[0.4,-1,1],B[-0.6,-1,1],C[0.2,-1,1],D[0.5,-1,1],E[0.9,-1,1])");

  RF->LoadSpeciesPDF("FitFunc",3000);

  TChain *chainMC=new TChain("BinnedTree");
  chainMC->Add("studyRecBins/Bin1/TreeFitFunc.root");
  if(!PDF->SetEvTree(chainMC)) exit(0);
  //PDF->CheckIntegralParDep(2); //Check if integral constant
  //PDF->SetUseWeightsGen(kFALSE); //Use accept/reject not weights
  RF->TotalPDF();
  RF->SetStudyPDF("FitFunc");
  //RF->SetStudyPlot();
  RF->SetNStudyTrials(1);
  RF->StudyFit();
 }
```

Summary

- Full chain CLAS12 analysis is underway
- Use CLAS12 gemc/common tools to produce reconstructed particles
- Convert to ROOT
- Analyse 4-vectors etc.
- Perform fits to pseudo data
  - Extended maximum likelihood with acceptance correction
- Validate fits with ToyMC
- Code and documentation (almost) available

hsroot Enviroment

- Additional analysis classes loaded in ROOT and compiled with ACliC at run time
- There are 3 independent components
  - Simplify interface to PROOF and TSelector classes (hssel)
  - Simplify interface to RooFit (hsfit)
  - Control particular analysis with Projects (hsproj) and THSParticles
- An analysis may incorporate all 3 or any individual component within ROOT operation
- All source files stored in HaSpect dir

Running hsroot

- hslogon.C
  - can be used to define extra useful functions to be run in ROOT session
  - Interpret command line arguments
- hsroot Command line args

  --hssel load HSSelector classes

  --hsfit load HSRooFit classes

  --proof=N initialse proof for N workers

  --THSMyClass.C compile and load new class to be used in this session

  --hsin set input file directory

  optional

  --hsout set output destination

  Can be file or dir

  --macrodir=extradir add additional path to your class

  --makeall compile all classes

Commands

```
root --hssel
hsroot[0] ...do stuff with hssel classes

root --hsfit
hsfit[0] ...do stuff with hsfit classes

root --hssel --proof=3 MyMacro.C

...run Control.C with proof and hssel classes

root --hssel MyMacro.C --hsout=test.root
*
```

* Can use function hsout() in Control.C to set ouput file to test.root (similarily for hsin)

Can also set in shell via env variables e.g. setenv HSOUT /destination/

Or in MyMacro.C :Hout("/destination/");

Temporary COATJAVA interface

Lund file read by gemc

Gemc output in HIPO format

Event Reconstruction with coatjava, output HIPO

Use groovy to read HIPO convert to text
- -Rec::Particle, REC::Detector, FT::Particle
- -Loop through these banks write each track

Currently write :

4-vector : vertex : time : PID : charge : trackMass

Convert these into THSParticle class
Store as arrays of THSParticles for both rec and gen

Signal Selection with THSProject

Users derive their own ROOT C++ class

- Handles EVENTS
- Take a detected state of HSParticles
- Produce quantities for
  - Fitting: final 4-vectors, variables (angles,masses), discriminators (signal/back)
  - Diagnostics: compare simulated/real,...
- Define behaviour for all topologies
- Permutate all combinations
  - Call WorkOnEvent for each
  - Also works when PID not known, just charge
- Same code for simulated and real experiment

```
WorkOnEvent(){
fGoodEvent=kTRUE;
//Find the detected particles for event
FindTopology();
//Do they have a defined topology?
if(fCurrTopo==fTIDprot_pip_pim_pi0_omega)
Init_prot_pip_pim_pi0_omega();
  else
if(fCurrTopo==fTIDprot_pip_pim_pi0)
Init_prot_pip_pim_pi0();
  else if(fCurrTopo==fTIDprot_pip_pim)
Init_prot_pip_pim();
  else fGoodEvent=kFALSE;
//From here on same for each topology
//Calc kinematics
ProductionKinematics();
}
```

Define topology

- How different final state particles are reconstructed will differ for different detected final states
- Handle via topology Init functions
- THSProject determines topology for event
- User codes behaviour

```
void THSPhotoOmega::Init_prot_pip_pim(){
  //Fill our data member particles
  //User is responsible for indicing right
  fProton=(fVecProtons.at(0));
  fPip=(fVecPiPs.at(0));
  fPim=(fVecPiMs.at(0));
  //make pi0 missing particle
  fPi0.SetP4(fBeam+fTarget-fProton.P4()-
                  fPip.P4()-fPim.P4());
  fPi0.SetMeasMass(fPi0.P4().M());
  fPi0.TakePDGMass();
//Reconstruct Omega
  fOmega.SetP4(fPip.P4()+fPim.P4()
+fPi0.P4());
  fOmega.SetMeasMass(fOmega.P4().M());
  fOmega.TakePDGMass();
  //Reconstruct discriminators for this
topology
  fOmegaMass=fOmega.MassDiff();
  fMissMass=fPi0.MassDiff();
}
```

THSProject output tree

- User defines information to be saved in tree for fitting
- User takes care of tree filling/saving outwith project
- Pass project the tree before start loop

```
THSPhotoOmega::ProjectOutTree
                    (TTree* tree){
//fFinal=Final State HSParticles
tree->Branch("Final",&Final);
tree→Branch
("MissM",&fMissM,"MissM/D");
tree→Branch
("OmMass",&fOmMass,"OmMass/D");
tree→
Branch("Topo",&fCurrTopo,"Topo/I");
tree->Branch("t",&f_t,"t/D");
tree->Branch("W",&f_W,"W/D");
}
```

Simulated Events with THSProject

- Access true values from fGenParts
- Directly set recon particle truth
- THSProject attempts to determine if correct tracks reconstructed
- Sets “Correct” flag
- Can then determine resolution, combitorial background,...

```
//When only analysing generated
Init_Generated(){
fElectron=frGenParts->At(0);
//When analysing recon
fElectron.SetTruth(frGenParts→At(0))

WorkOnEvent(){
//Check if recon match gen
CheckTruth();

//Check difference in recon angle
fElectron.ResTheta();
...
```

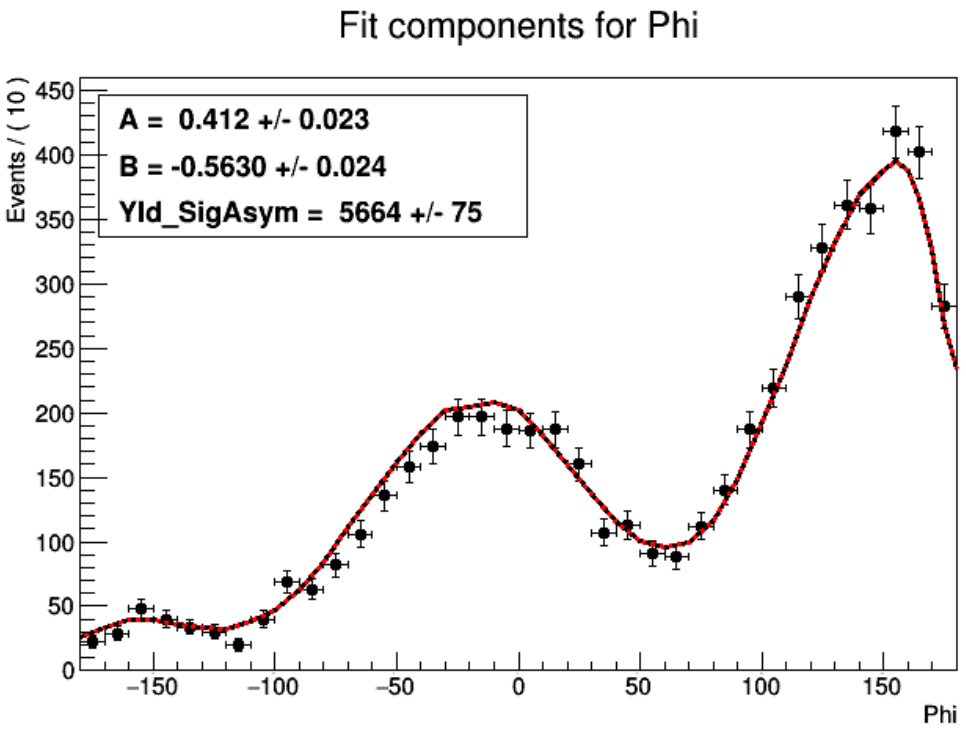
Looping Events with THSDataManager

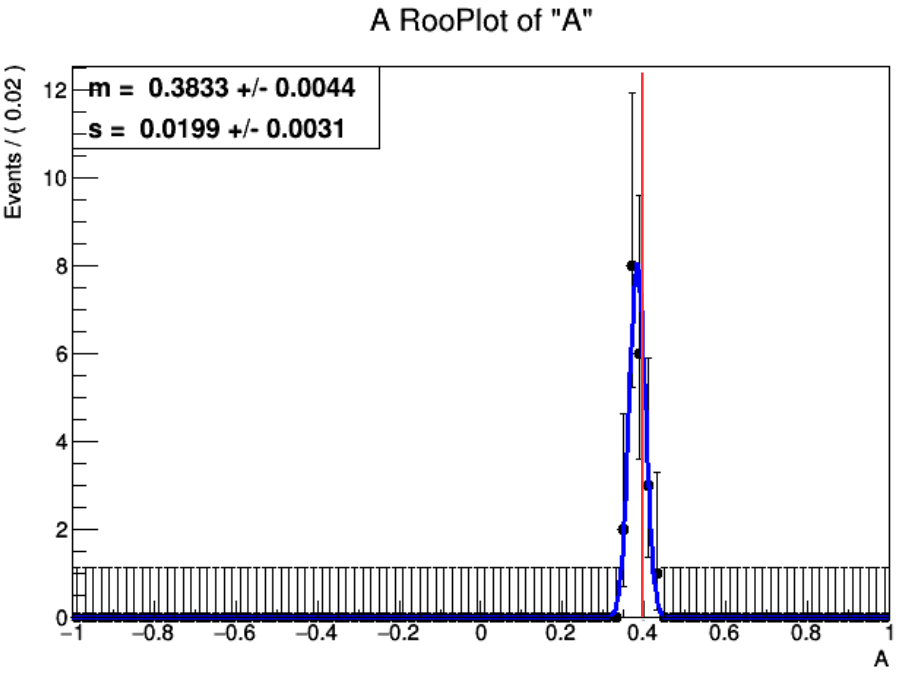
- Handled in ROOT script
- Declare project to handle events
- Declare data manager to handle data files
- Loop over events

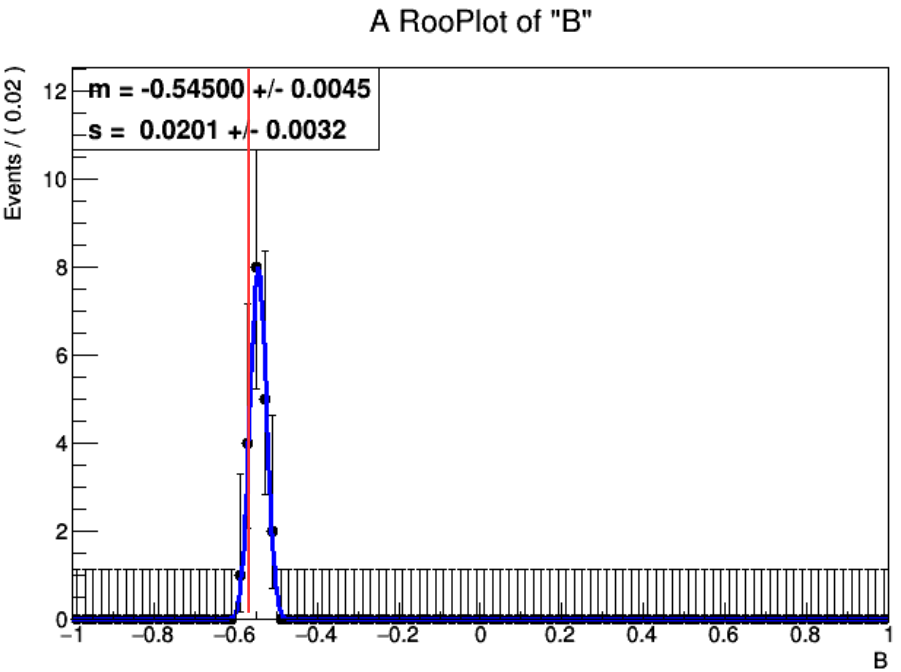
```
//Create Project
THSPhotoOmega* PO=new THSPhotoOmega();
THSDataManager* dm=new THSDataManager();
dm→InitTreeReader
("Generated.root","HSParticles");
//connect Project to HSParticles
PO->SetDetParts(
     dm->GetReaderParticles());

while(dm->ReadEvent()){
    PO->WorkOnEvent();
  }
```

Res 10 Asymmetric acceptance

Include acceptance in Likelihood normalisation







With Corrections applied:

A = 0.441153 +/- 0.0234849
B = -0.58103 +/- 0.0236717