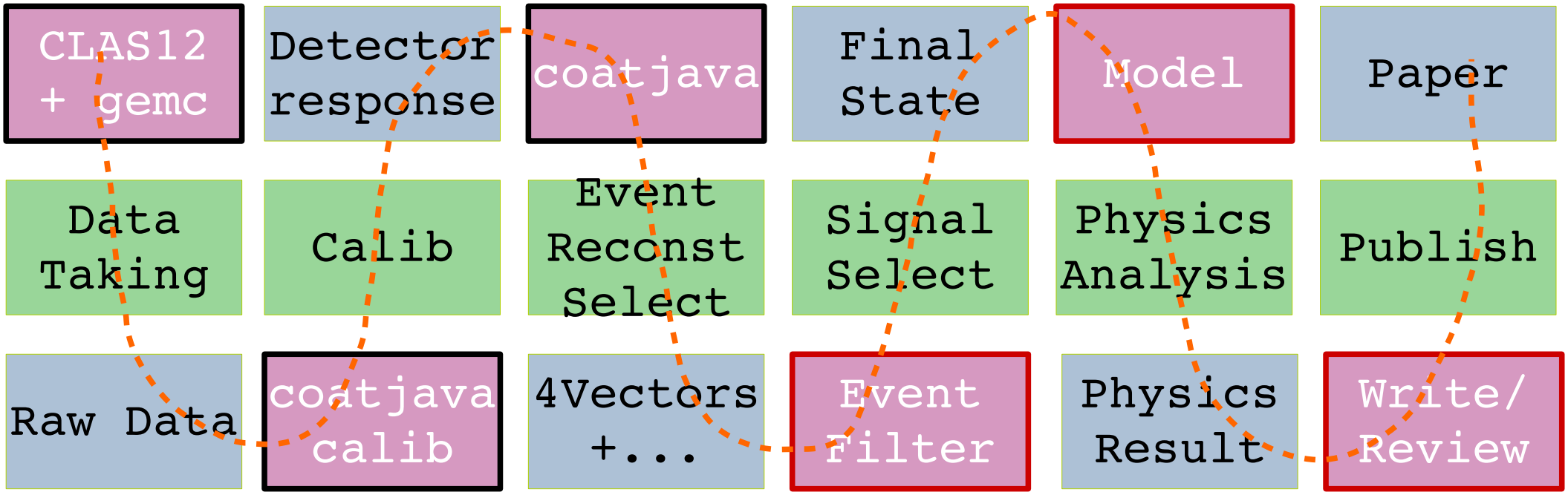
Signal Selection and Physics Analysis Tools

Hadron Spectroscopy Working Group

Derek Glazier
University of Glasgow
(HASPECT working group)

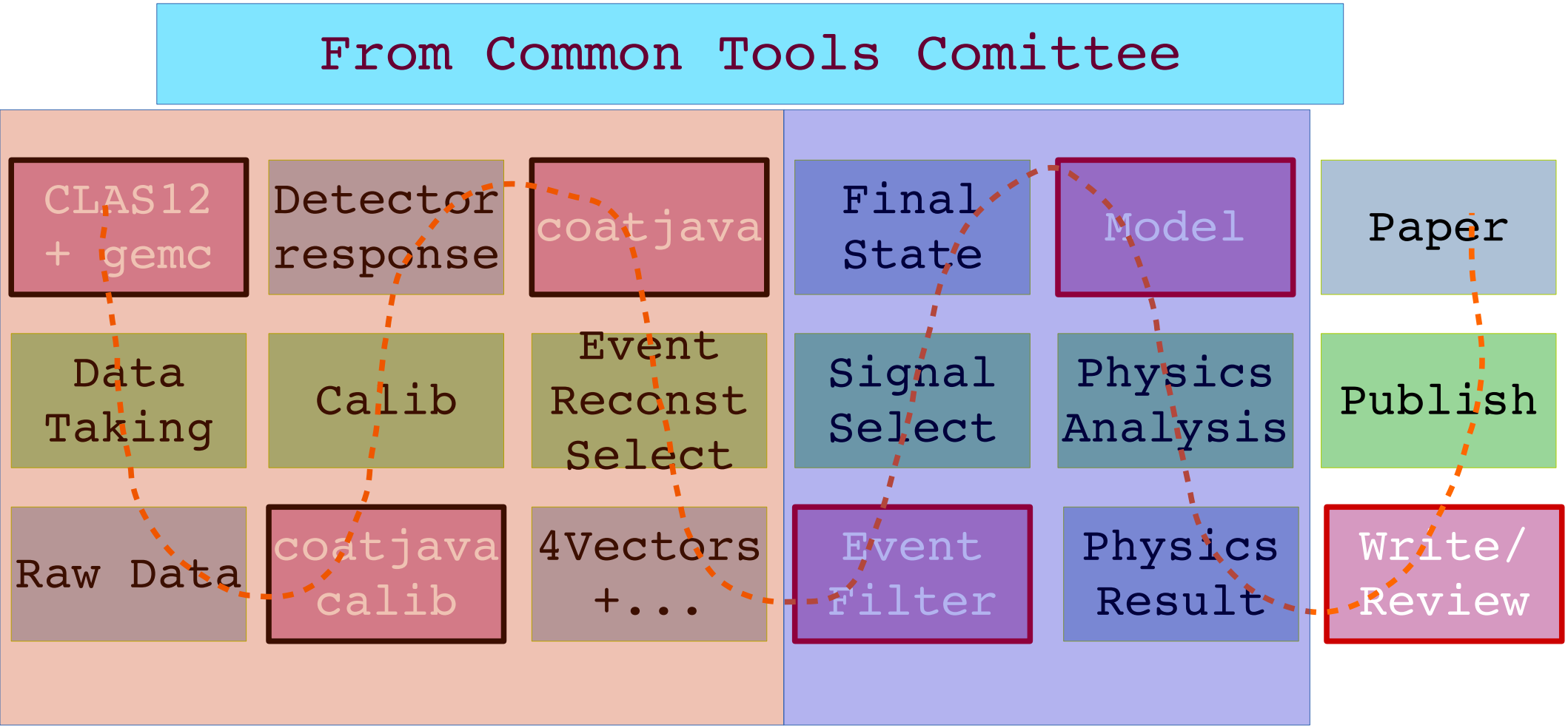
Route to Publication - MesonEx



6 Month 6Month 3Month 3Month 12Month 6Month

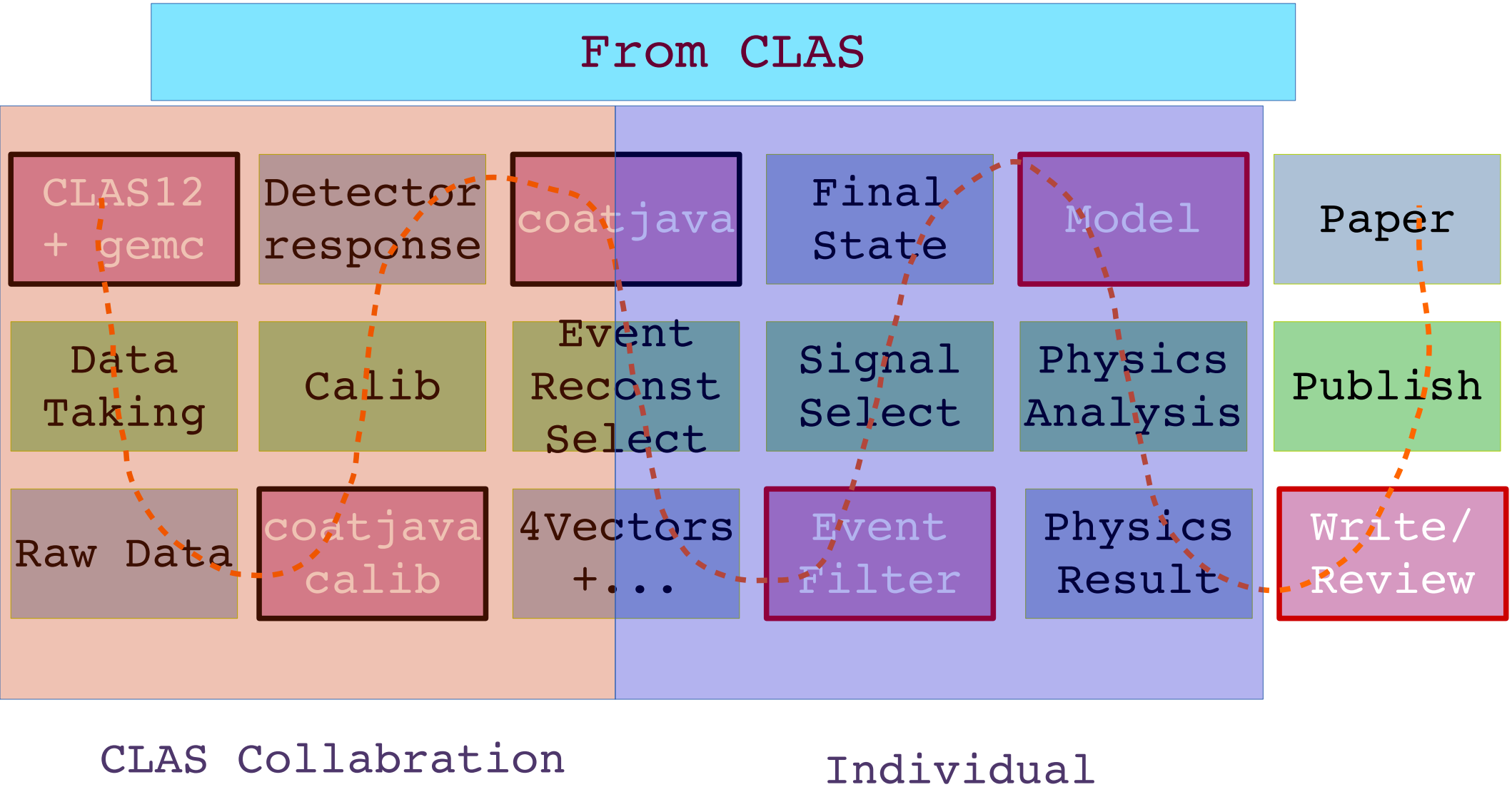
Possible ~ 3.5 years

We would like fast and reliable publications



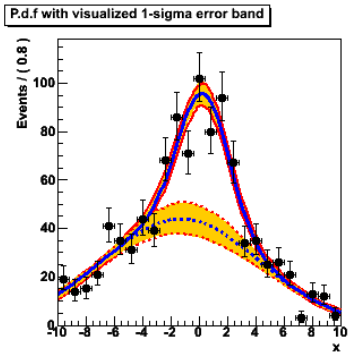
CLAS Collaboration
Run Group

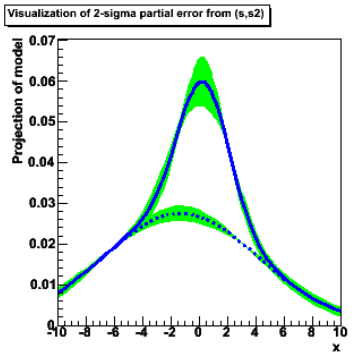
Open
Individual?
Experiment?
Working Group?
…
But with “guidance”


From CLAS
CLAS12 + gemc
Detector response
coatjava
Final State
Model
Paper
Data Taking
Calib
Event Reconst Select
Signal Select
Physics Analysis
Publish
Raw Data
coatjava calib
4Vectors +...
Event Filter
Physics Result
Write/ Review
CLAS Collabration
Individual

What is an analysis framework ?

RooFit





COATJAVA → Calibration
Event Reconstruction
Event Selection
…

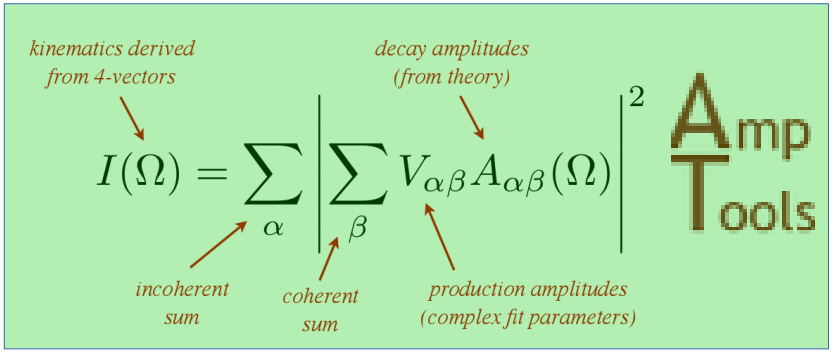
We should build on these producing code for specific Analysis tasks.

What types of analysis are we performing?

Model Dependent : We must interface to these tools

The PyPWA Project

$$I(\tau) = \sum_{i,j} \sum_{b,b'} {}^{i}A_b(\tau)\ {}^{i,j}\rho_{b,b'}\ {}^{j}A^*_{b'}(\tau).$$



Model Independent :

Peak Hunting

Cross Sections

Spin Density Matrix Elements

Polarisation Observables

Signal Selection :

Q-Factor

sPlots

Sideband Subtraction

TMVA Cut

Missing Mass Cut

Colours imply use of Maximum Likelihood

Model Dependent :

The PyPWA Project

$$I(\tau) = \sum_{i,j} \sum_{b,b'} {}^{i}A_b(\tau)\ {}^{i,j}\rho_{b,b'}\ {}^{j}A^*_{b'}(\tau).$$

kinematics derived from 4-vectors

decay amplitudes (from theory)

$$I(\Omega) = \sum_{\alpha} \left| \sum_{\beta} V_{\alpha\beta} A_{\alpha\beta}(\Omega) \right|^2$$

incoherent sum

coherent sum

production amplitudes (complex fit parameters)

AmpTools

Model Independent :

Peak Hunting

Cross Sections

Spin Density Matrix Elements

Polarisation Observables

Signal Selection :

Q-Factor

sPlots

Sideband Subtraction

TMVA Cut

Missing Mass Cut

In particular high dimensionality of final states demands event-by-event maximum likelihood

Why RooFit?

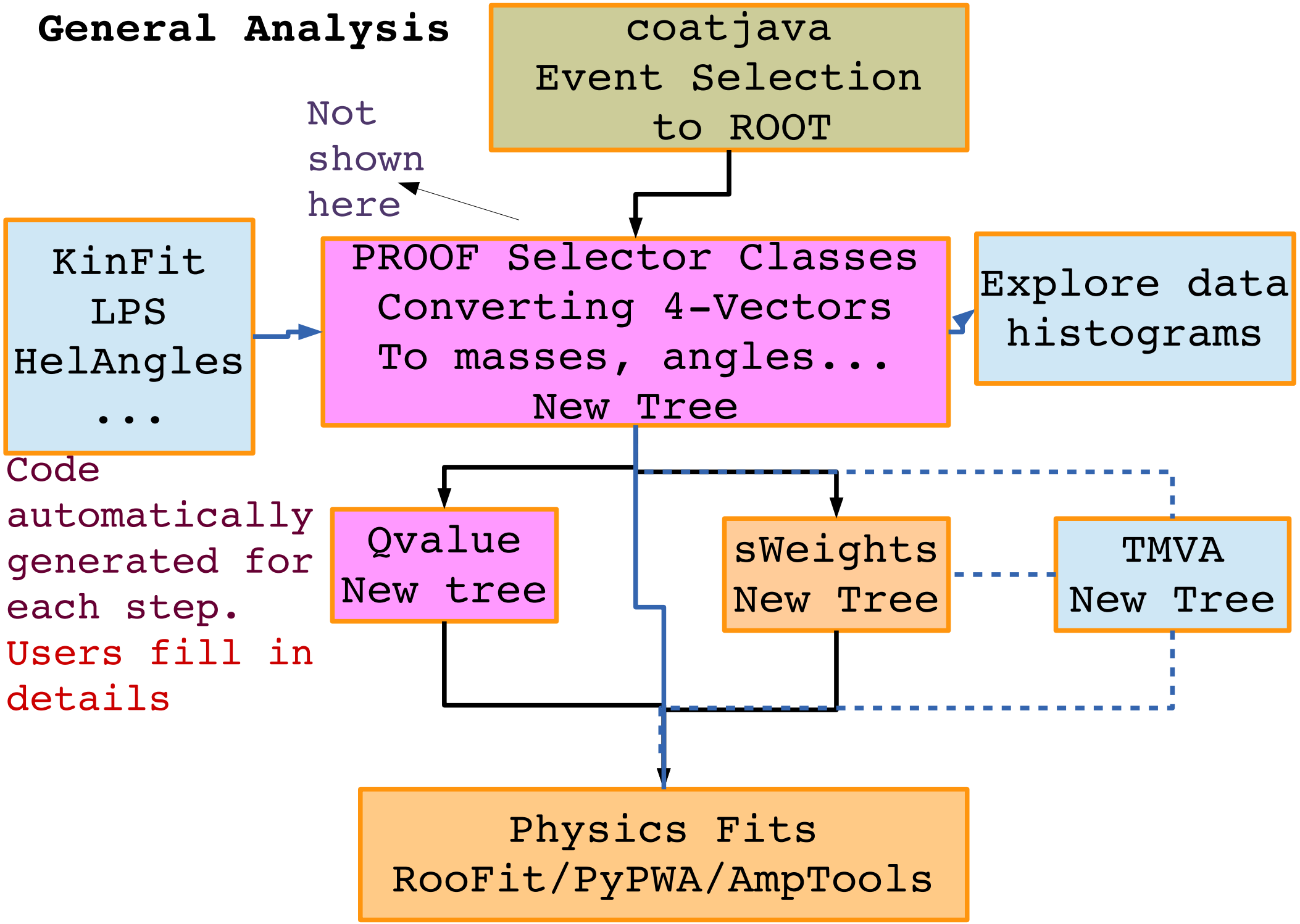
Many issues arise

- Standard ROOT function framework clearly insufficient to handle such complicated functions → must develop new framework
- Normalization of p.d.f. not always trivial to calculate → may need numeric integration techniques
- Unbinned fit, >2 dimensions, many events → computation performance important → must try optimize code for acceptable performance
- Simultaneous fit to control samples to account for detector performance

Why THSRooFit?

- Make RooFit even simpler for HASPECT User
- Little overhead for performing fits in many bins
  - Includes correct handling simulated models
- Straightforward handling of Weights
- Same code for sWeights and Observables

General Analysis



Signal Selection : sWeights

M. Pivk,F.R. Le Diberder,Nucl.Inst.Meth.A 555, 356-369, 2005

Given discriminatory PDF for signal and background calculates weight :

$$ {}_s\mathcal{P}_n(y_e) = \frac{\sum_{j=1}^{N_s} \mathbf{V}_{nj} f_j(y_e)}{\sum_{k=1}^{N_s} N_k f_k(y_e)} $$

N_s = Number of species

f_k = PDF for species k

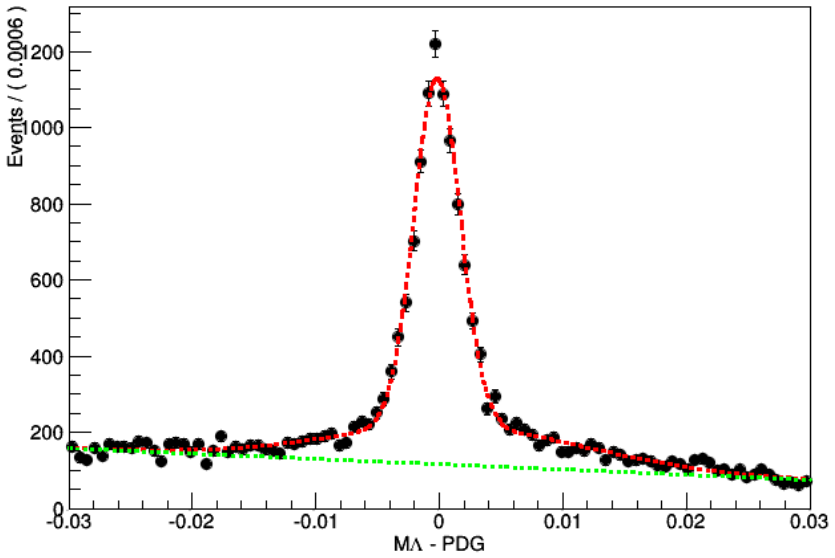
N_k = Yield for species k

V = covariance matrix

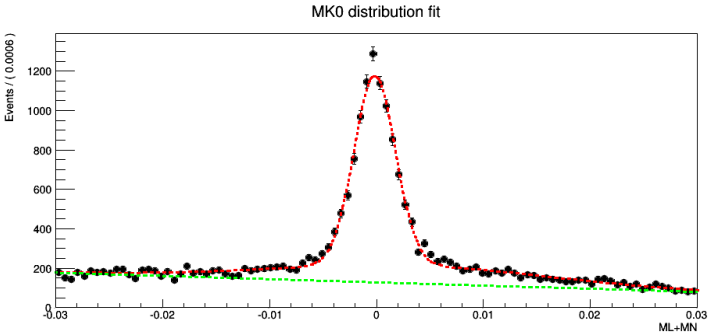
Part of RooStats(used here)

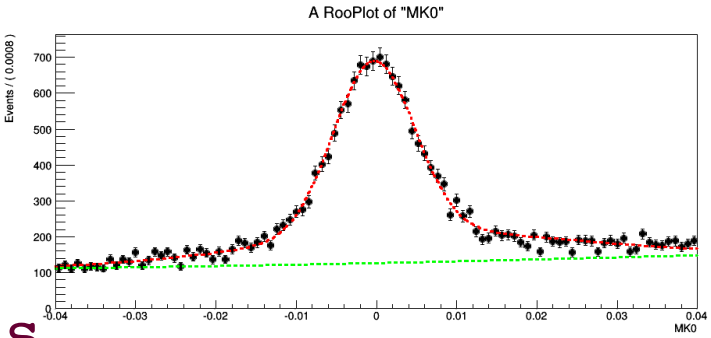
Can include multiple signal and background species

Can fit multidimensional discriminatory PDF



Only as good as fit model...

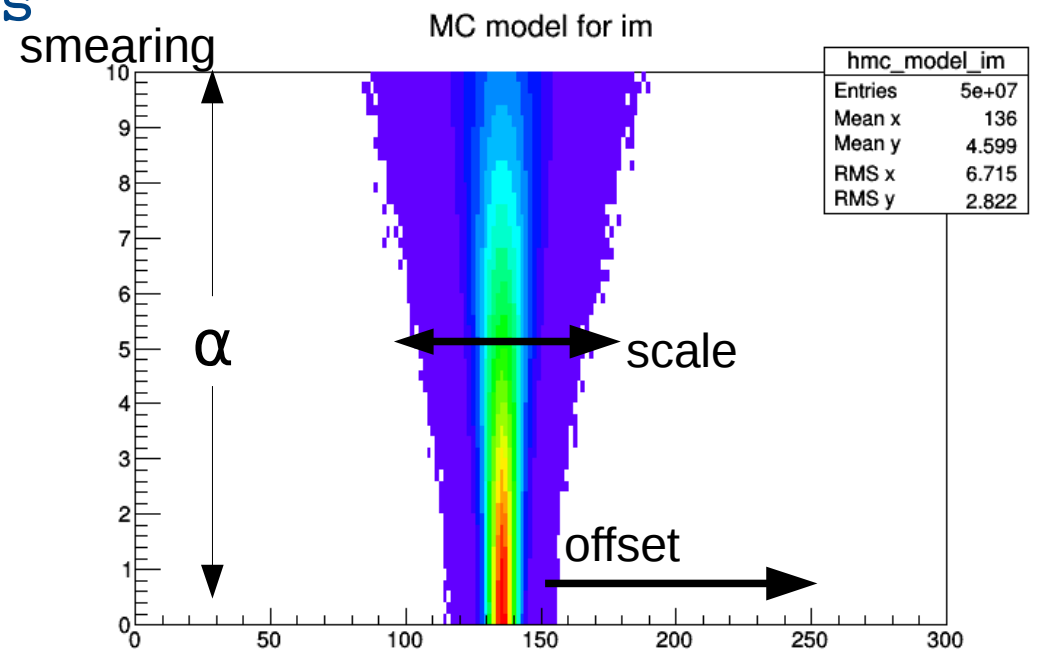




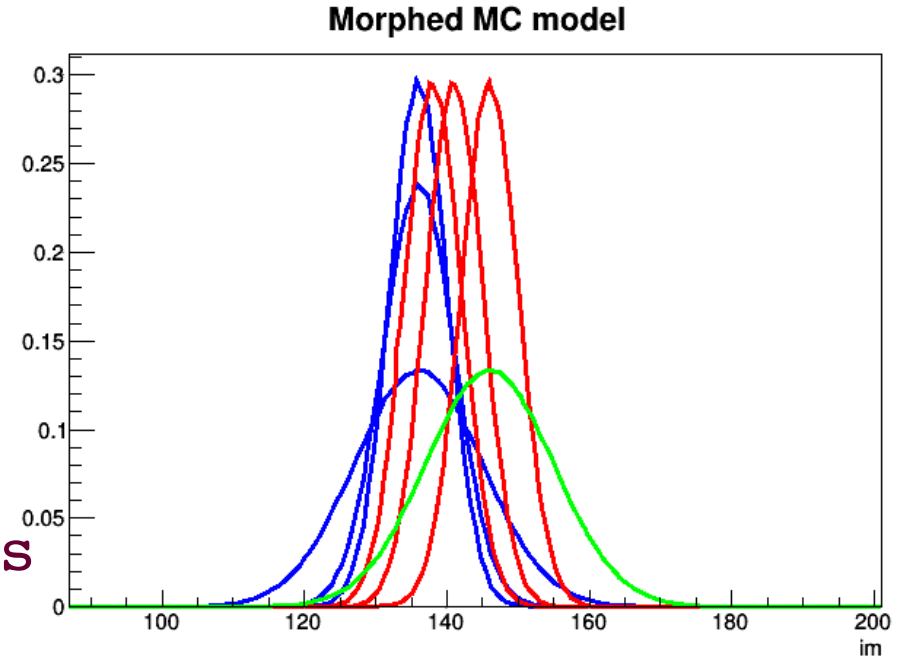
Can use directly in likelihood fits

Signal Selection : Simulated Models

Signal shapes are not always well described by parameteric functions
⇒Simulated PDFs
systematic uncertainty in shape accounted for via morphing with additional nuisance parameters



Construct new RooFit PDF
Supply simulated events
Sequential 1D histograms
Smoothed and interpolated
Adding greater additional smearing with morphing parameter α
Additional offset,scale
Constrain these with Gaussians

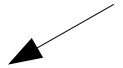


Sample ROOT macro :

Note all classes compiled at run time Using ROOT ACLiC

```
THSEventsFit* RF=new THSEventsFit("SF");  //Manager class
RF->LoadVariable("Mmiss[-0.1,0.15]");//Discriminatory variable
RF->LoadBinVars("Eg",10,3,4);//Split into 10 Egamma bins
RF->LoadBinVars("t",5,0,4);//Split into 5 t bins

////////////////////////////Make Model Signal
RF->Factory("THSEventsPDF::Signal(Mmiss,alpha[2,0,4],off[0,-5,5],scale[1,09,1.1])");
TChain chainmcL("HSParticles","mcsignal");
chainmcL.Add("mc_ppip_cor.root");
//add mc data to make model
((THSMorphPDF*)RF->GetWorkSpace()->pdf("Signal"))\\
                    ->AddSmearedModel(&chainmcL,RF->GetAuxVars());
////////////////////////////Make BG model (same code again)
...
RF->LoadSpeciesPDF("Signal");
RF->LoadSpeciesPDF("BG");
RF->TotalPDF();//Total PDF
////////////////////////Load Data
TChain chain("HSParticles");
chain.File("twopi_ppip_pmiss.root");
RF->LoadDataSet(&chain);//import to RooFit

////////////////////////Fit Model to data
RF->RunWeights(10); //Run 10 fits and use the best for weights

RF->DrawTreeVar("Mpipm",200,0,2);//Draw sum weighted variables
RF->DrawTreeVar("MmissP",200,0,2);

RF->SaveWeights(); //save Weights to file for further analyses
```

Parameter ranges



Only really need to configure variable and file names

Useful Class THSBins

Can split input data into N dimensional bins and peform fit for each bin

```
RF->LoadBinVars("Eg",10,3,4);
Double_t tbins[]={0.1,0.2,0.3,1,2,5};
RF->LoadBinVars("t",5,tbins);
```

Bin number stored as a branch in an associated tree. This tree is used to filter events for individual fits

Can be used to create persistant RooFit Workspace, containing data and PDF for each bin.

```
RF->PrepareForFarm();
```

Workspaces can be sent to farm for faster processing

Useful Class THSWeights

Container for sWeights (or any other weight)

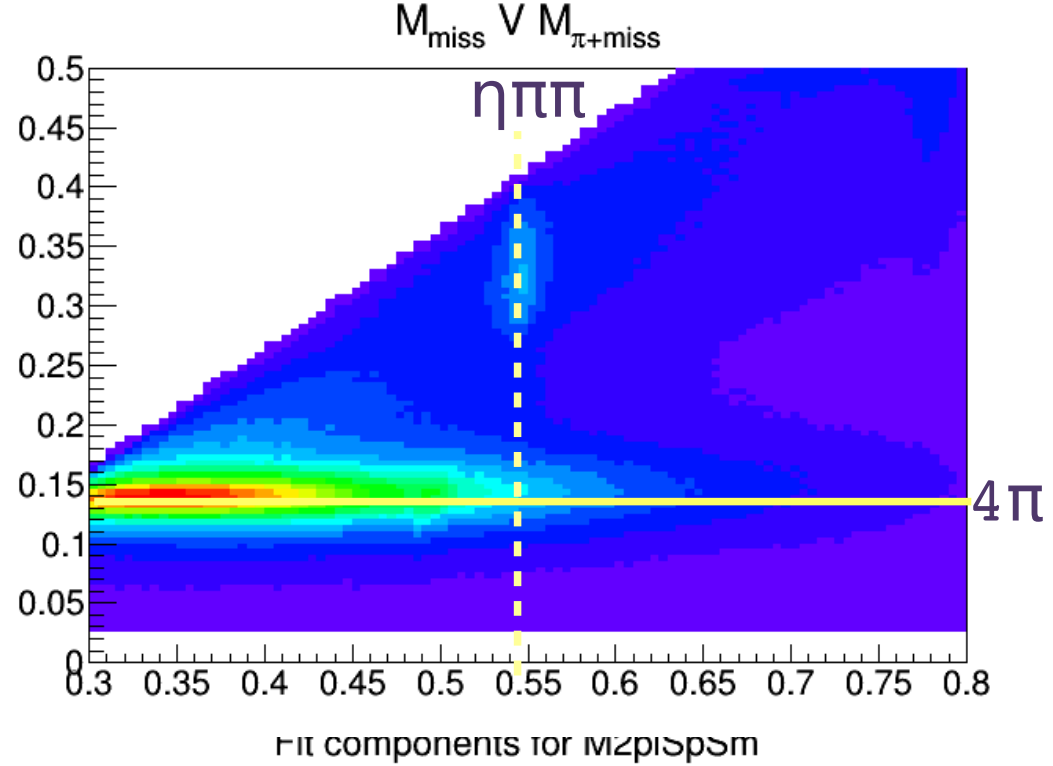
Requires global event ID to synchronise with events
(Passed from THSRooFit)

Stored in TTree with branch names = species
(e.g. “Signal”, “Background”)

```
THSWeights* wts=new THSWeights("TotalWeights");
//If fits done on farm merge all bins
wts->Merge("WeightsEg","WeightsAll.root");
//Loop over data get current event ID
wts->GetEntryBinarySearch(ID);//Get Weight for ID
Weight = wts->GetWeight("Signal");
```

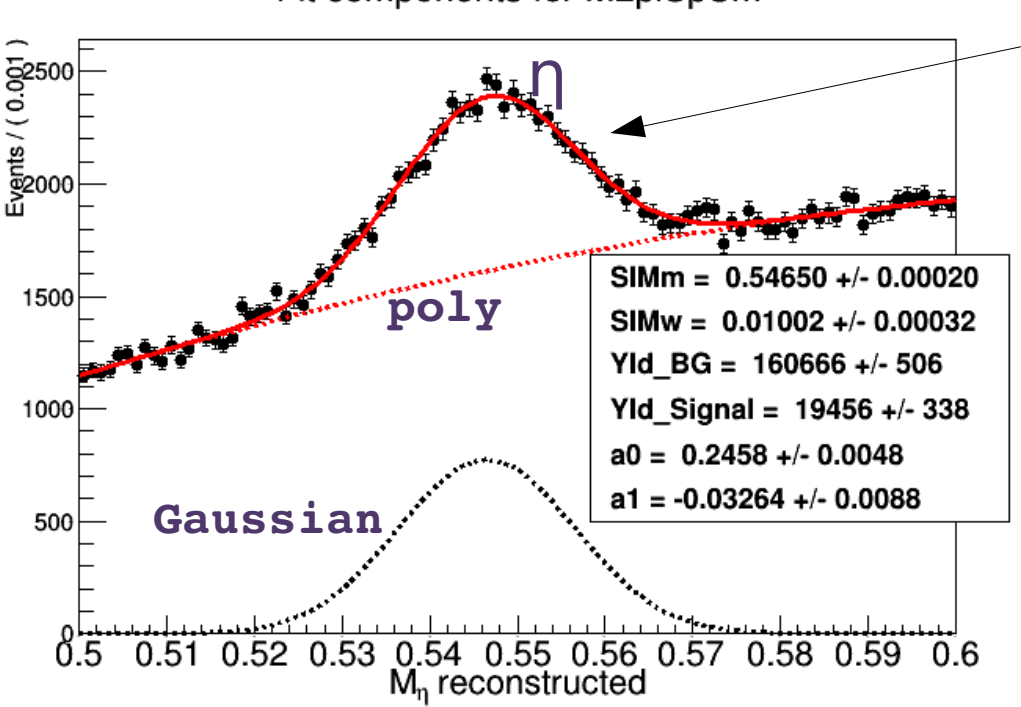
Weights can also used for Acceptance,
Simulation matching (with data),
Simulation modelling (with model)

Case Study : Bump Hunting ηππ

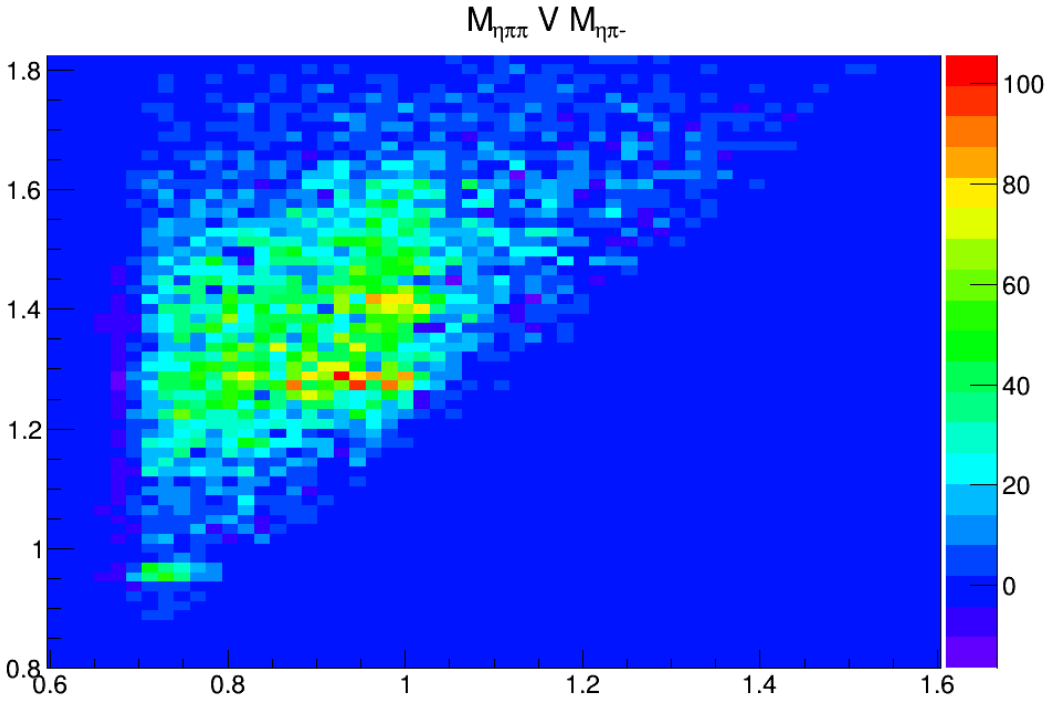


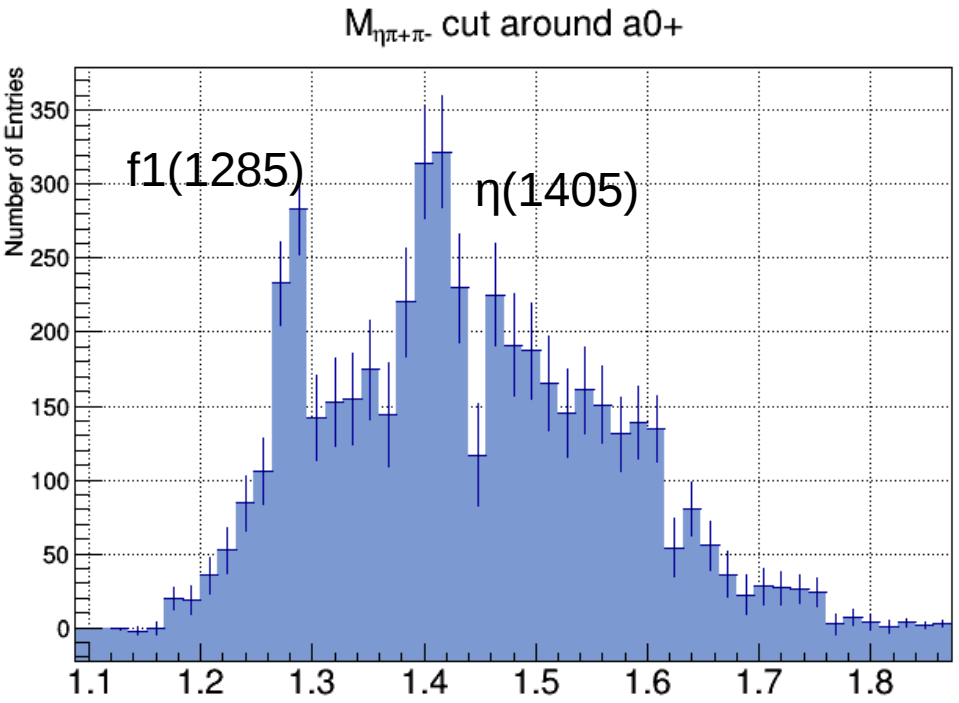
Use g11 data set
from n10 skim
Detect p,π+,π+,π-
Missing π-π0

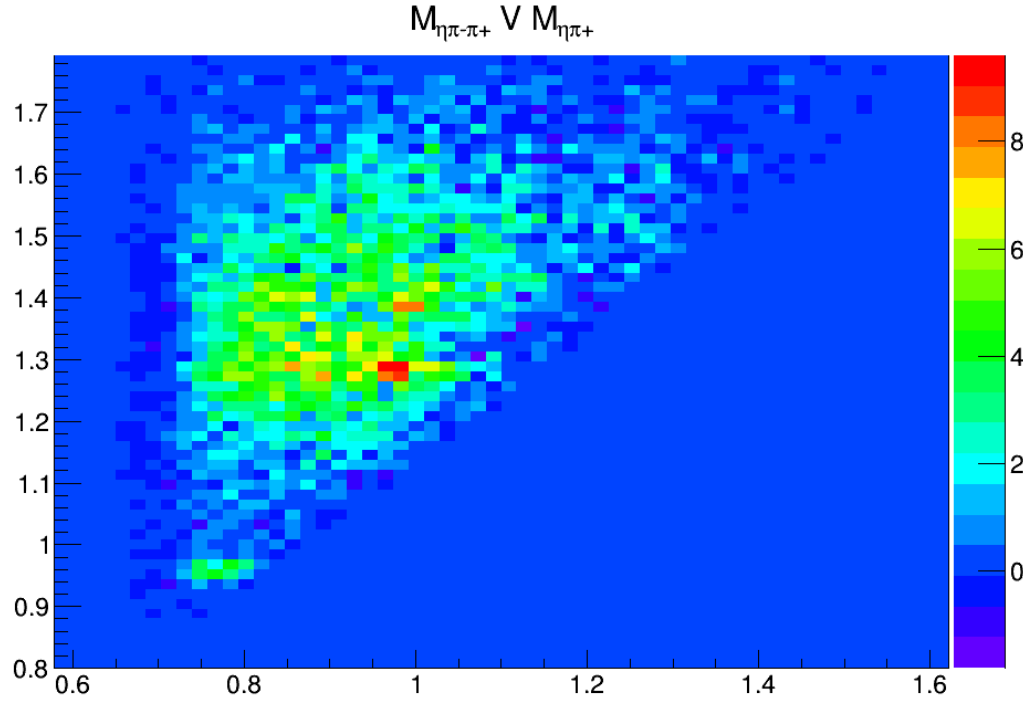
Fit this to produce
sWeights for ηππ

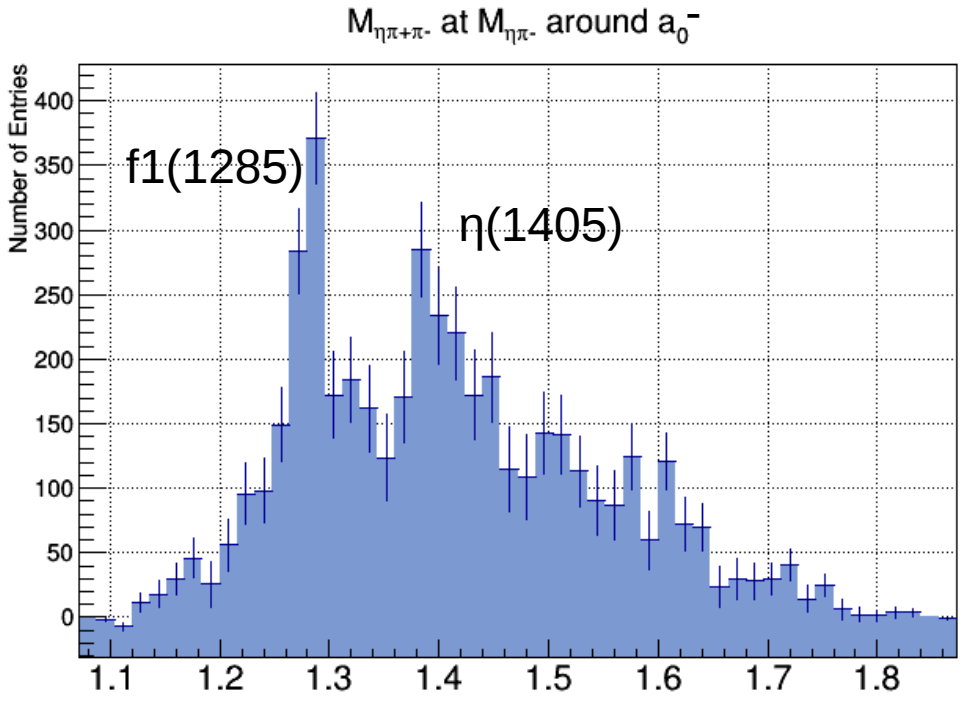


Cut on Longitudinal
Phase Space :
ηππ forward in CM
Proton backwrd


$M_{\eta\pi\pi}$ V $M_{\eta\pi^-}$
1.8
1.6
1.4
1.2
1
0.8
0.6
0.8
1
1.2
1.4
1.6
100
80
60
40
20
0

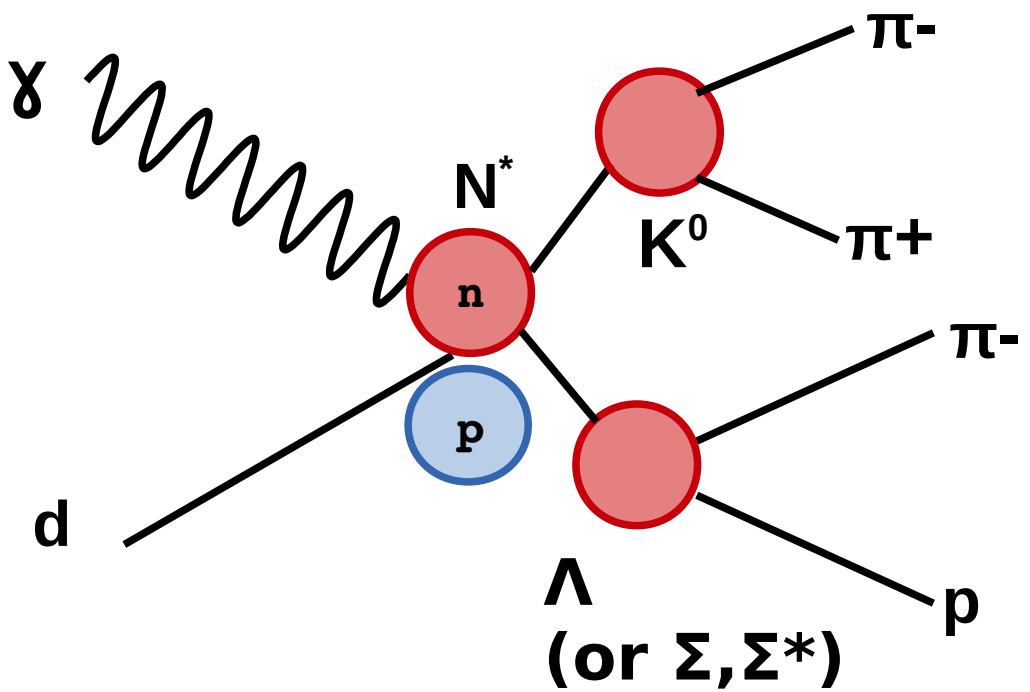

$M_{\eta\pi^+\pi^-}$ cut around a0+
Number of Entries
f1(1285)
η(1405)
350
300
250
200
150
100
50
0
1.1
1.2
1.3
1.4
1.5
1.6
1.7
1.8


$M_{\eta\pi^-\pi^+}$ V $M_{\eta\pi^+}$
1.7
1.6
1.5
1.4
1.3
1.2
1.1
1
0.9
0.8
0.6
0.8
1
1.2
1.4
1.6
8
6
4
2
0


$M_{\eta\pi^+\pi^-}$ at $M_{\eta\pi^-}$ around a_0^-
Number of Entries
f1(1285)
η(1405)
400
350
300
250
200
150
100
50
0
1.1
1.2
1.3
1.4
1.5
1.6
1.7
1.8

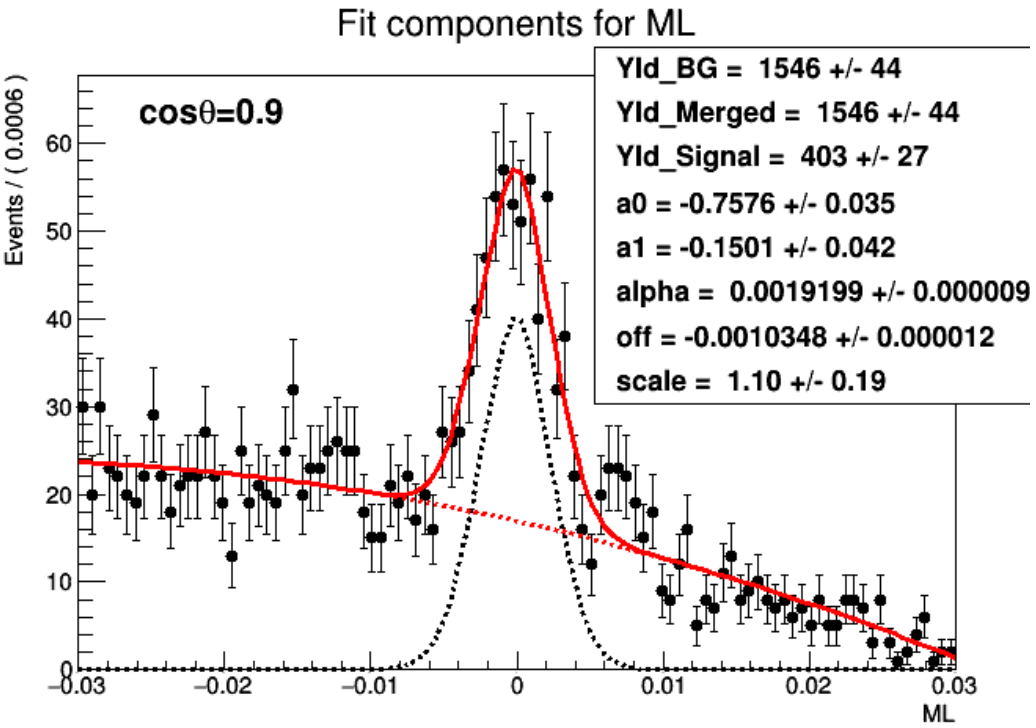
Case Study:K0 Photoproduction on neutron(Deuteron)

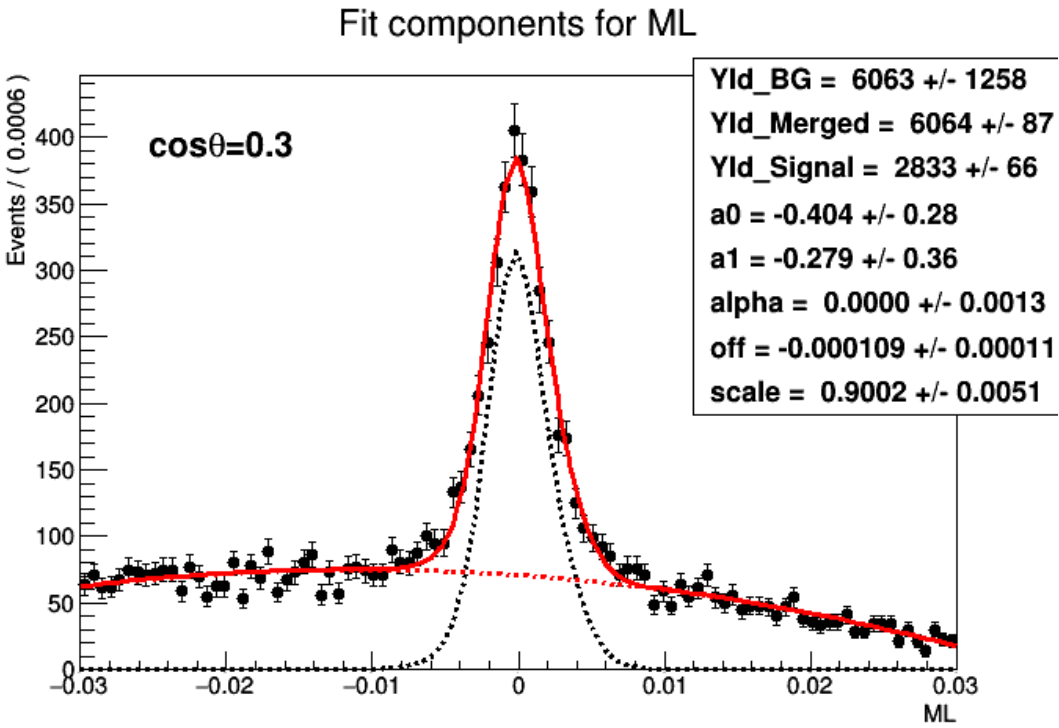
g13b; Event Selection with ROOTBEER; data for 1.3GeV Coherent Peak

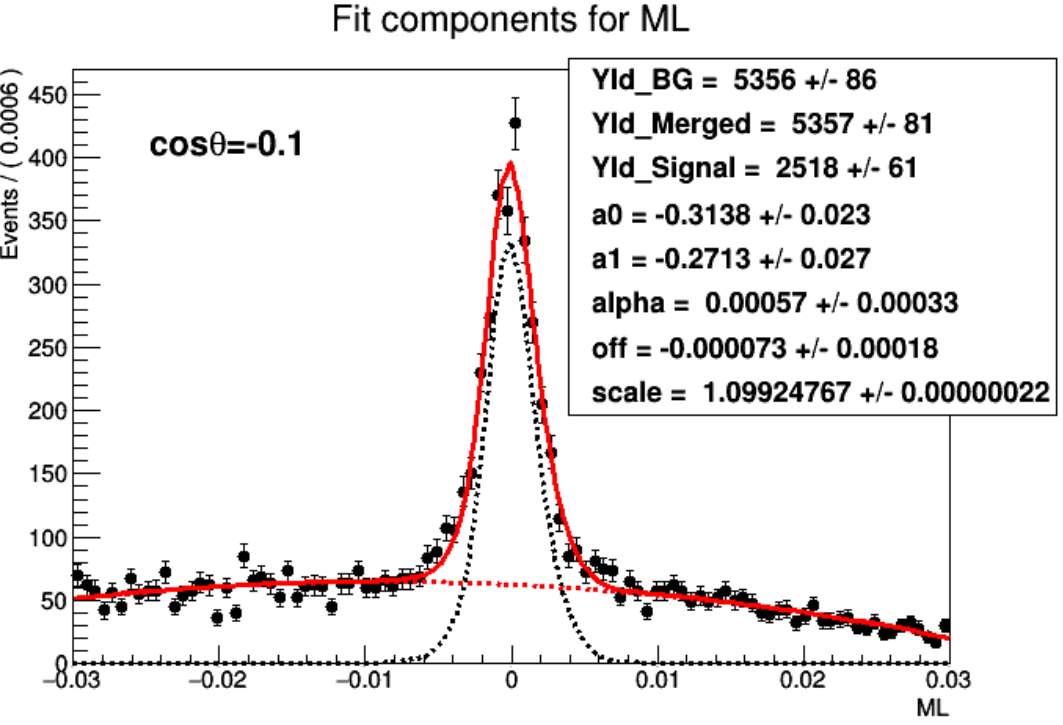


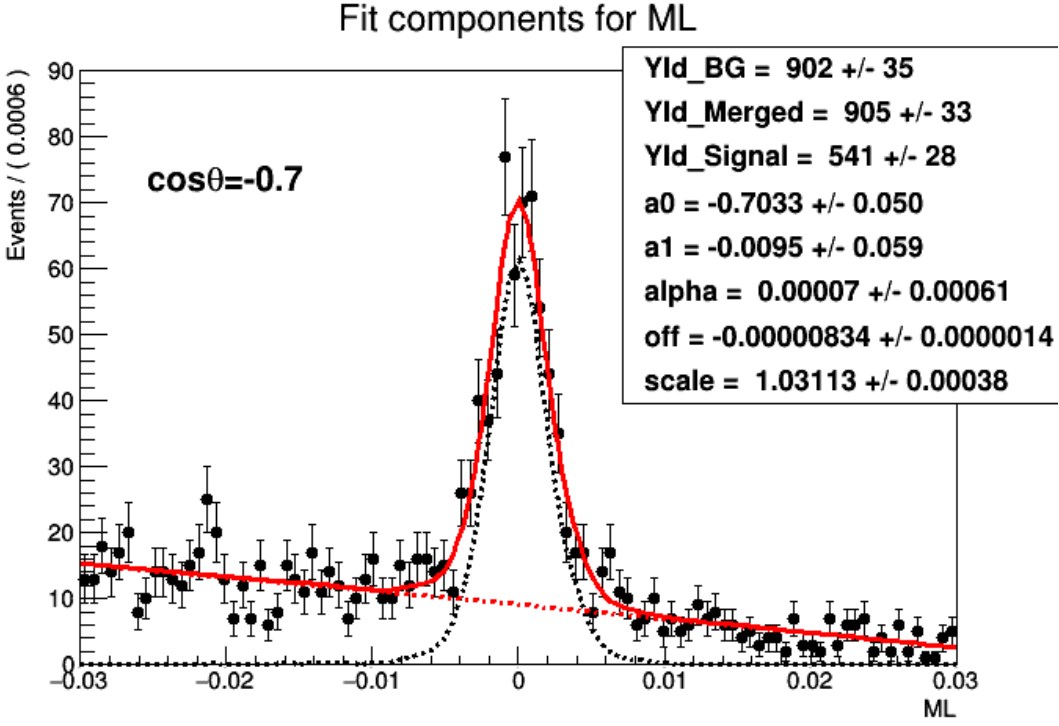
$$\frac{d\sigma}{d\Omega} \equiv \sigma\left(\phi, \cos\theta_x, \cos\theta_y, \cos\theta_z\right) = \sigma_0 \left\{1 - P_L^\gamma \Sigma \cos 2\phi + \alpha \cos\theta_x P_L^\gamma O_x \sin 2\phi + \alpha \cos\theta_y P - \alpha \cos\theta_y P_L^\gamma T \cos 2\phi + \alpha \cos\theta_z P_L^\gamma O_z \sin 2\phi\right\},$$

Select Event with a Lambda (Includes Σ,Σ*) Inv Mass pπ-

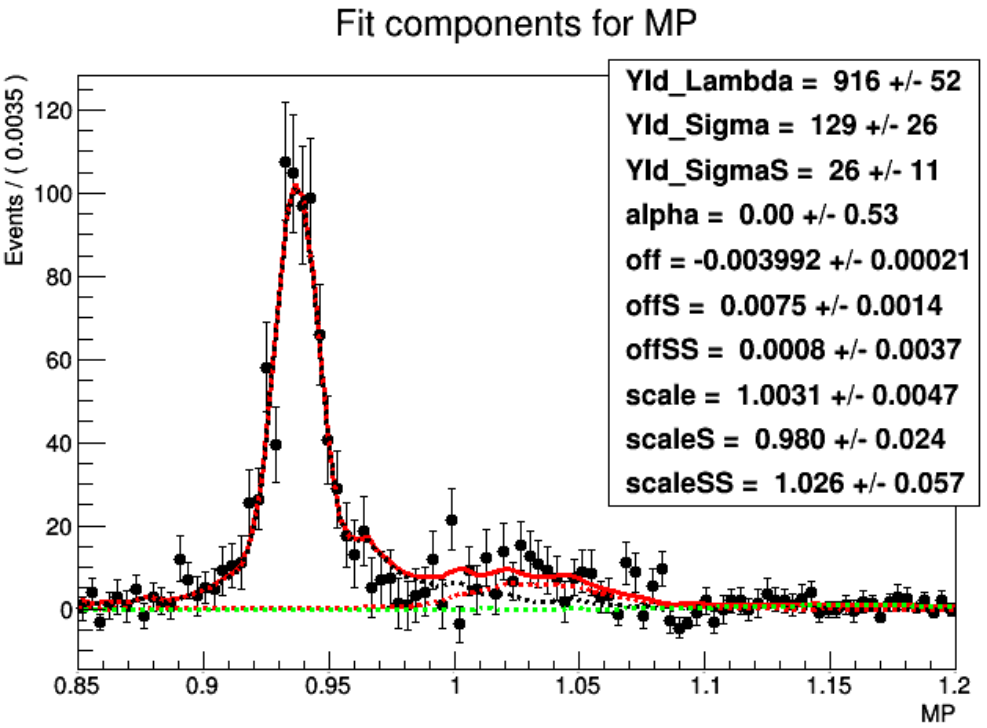


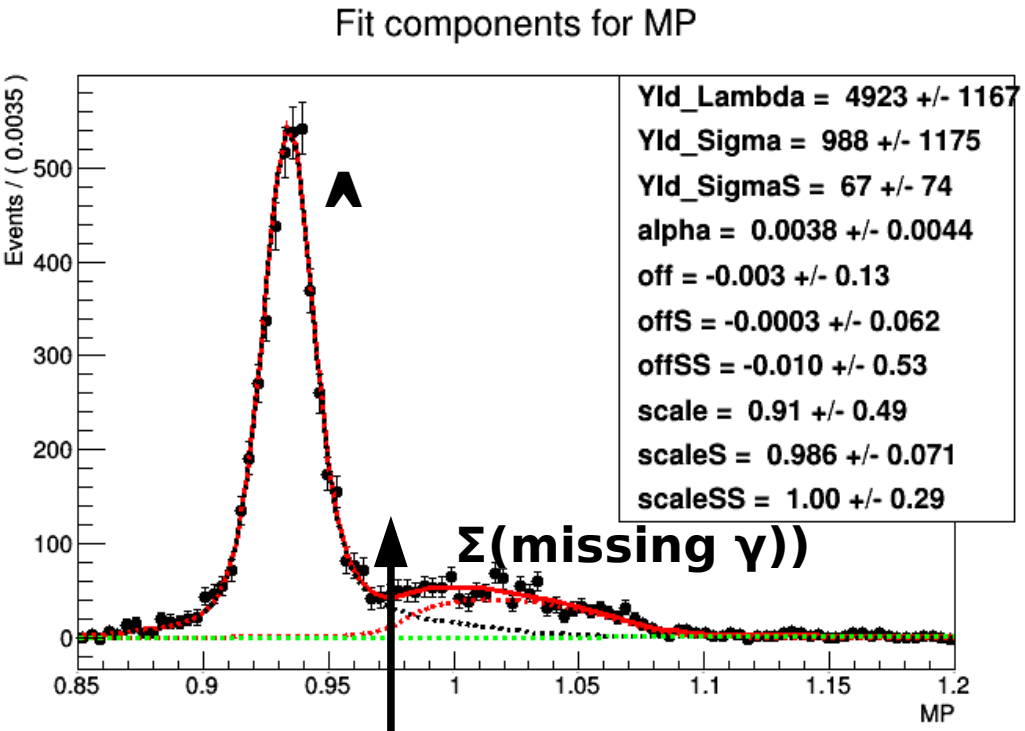


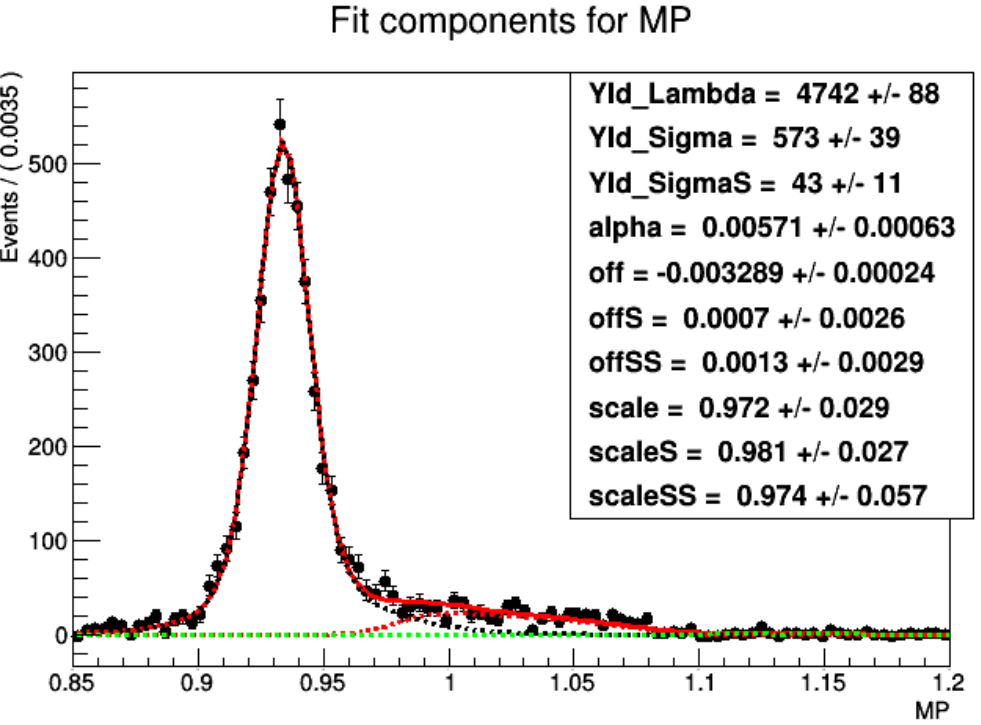


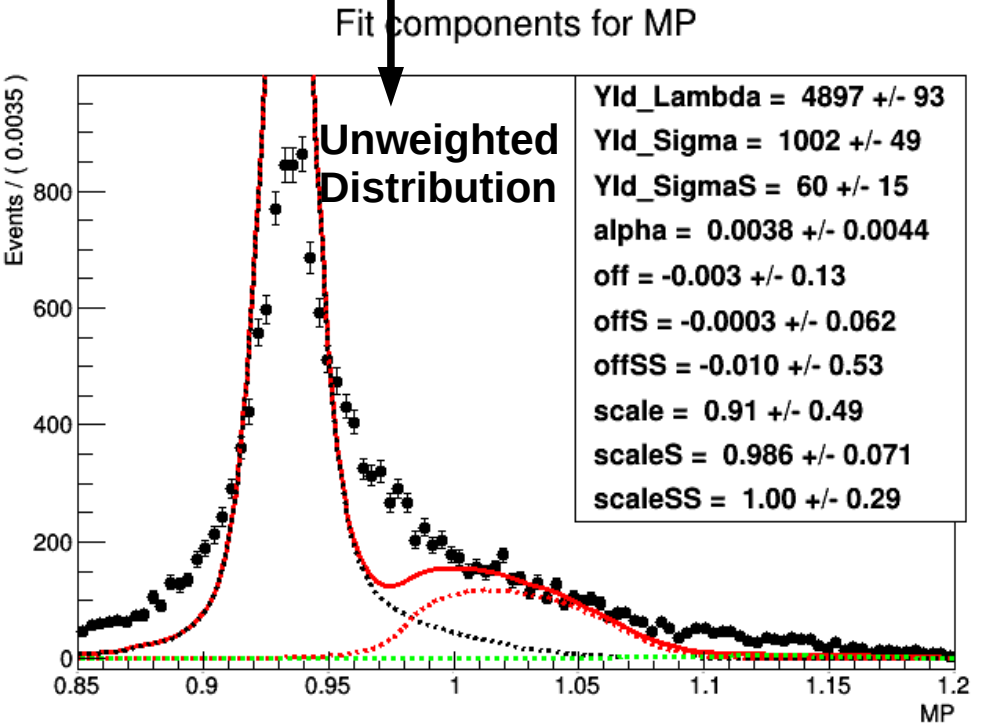


Select K0 Lambda, Missing Mass off 3πp (=spectator)









Pseudoscalar Meson Photoproduction(see g8b KΛ)

$$\frac{d\sigma}{d\Omega} \equiv \sigma\left(\phi, \cos\theta_x, \cos\theta_y, \cos\theta_z\right) = \sigma_0 \left\{1 - P_L^\gamma \Sigma \cos 2\phi \right.$$
$$+\alpha \cos\theta_x P_L^\gamma O_x \sin 2\phi$$
$$+\alpha \cos\theta_y P - \alpha \cos\theta_y P_L^\gamma T \cos 2\phi$$
$$\sigma(\xi) = \sigma_0\left(f - P_L^\gamma g\right) \qquad \left. +\alpha \cos\theta_z P_L^\gamma O_z \sin 2\phi \right\},$$

$$f(y) \equiv f = 1 + \alpha y P$$

$$g(\phi, x, y, z) \equiv g = (\Sigma + \alpha y T)\cos 2\phi - \alpha\left(x O_x + z O_z\right)\sin 2\phi,$$

$$A(\xi) = \frac{\sigma_\perp(\xi) - \sigma_\parallel(\xi)}{\sigma_\perp(\xi) + \sigma_\parallel(\xi)}, \qquad A(\xi) = \frac{P_L^\gamma g}{f}.$$

Physics Asymmetry : $$A(\xi) = \frac{P_L^\gamma g}{f}.$$

Measured asymmetry depends on relative luminosity :

$$\Delta L = \frac{L_\perp - L_\parallel}{L_\perp + L_\parallel}; \qquad a = \frac{\Delta L + A}{1 + +A\Delta L}.$$

$$\mathcal{P}\left(N_\perp, N_\parallel \mid a\right) = \frac{1}{Z}(1+a)^{N_\perp}(1-a)^{N_\parallel},$$

$$\mathcal{P}(\{E_i\} \mid \mathcal{O}, \lambda) = \prod_i \mathcal{P}_i(E_i \mid \mathcal{O}, \lambda)$$

No acceptance correction required!

Write a function....

```
Double_t THSPolObsPDF::evaluate() const
{
  // ENTER EXPRESSION IN TERMS OF VARIABLE ARGUMENTS HERE

  Double_t alpha=0.67;

  Double_t g=(B+alpha*ry*T)*TMath::Cos(2*(phi-delPhi))-
    alpha*(rx*Ox+rz*Oz)*TMath::Sin(2*(phi-delPhi));
  Double_t f=1+alpha*ry*R;

  Double_t As=Pg*g/f;

  Double_t a = (delLum+As)/(1+As*delLum);

  return 1+PS*a;
}
```

Sample ROOT macro for Observable fitting :

```
THSPolObsFit* RF=new THSPolObsFit("SF");  //Manager class
RF->LoadVariable("Phi[-180,180]");//azimuthal reaction plane
RF->LoadVariable("CosThx[0,180]");//Lambda decay angle
RF->LoadVariable("CosThy[0,180]");//Lambda decay angle
RF->LoadVariable("CosThz[0,180]");//Lambda decay angle
RF->LoadVariable("Plin[-1,1]");//Linear Polarisation
RF->LoadVariable("PS[-1,1]");//Linear Polarisation State (Para or Perp)
RF->LoadBinVars("Eg",6,1,2.4);//split into 6 Egamma bins
RF->LoadBinVars("Theta",10,0,180);//split into 10 Theta bins

////////////////////////////Make Beam Recoil PDF
RF->Factory("THSPolObsPDF
                  ::Klambda(Phi,CosThx,CosThy,CosThz,Plin,PS,Sigma,Ox,Oz,T,DelLum)");
RF->LoadSpeciesPDF("Klambda");

////////////////////////////Load Data
TChain chain("HSParticles");
chain.Add("data.root");
RF->LoadDataSet(&chain);//import to RooFit

//Weight data for K0Lambda
RF->LoadWeights("out/WeightsTotal.root","HSsWeights");
RF->SetWeightName("Lambda");

////////////////////////////Fit Model to data
RF->PrepareForFarm(10);
OR RF->FitMany(10);
```

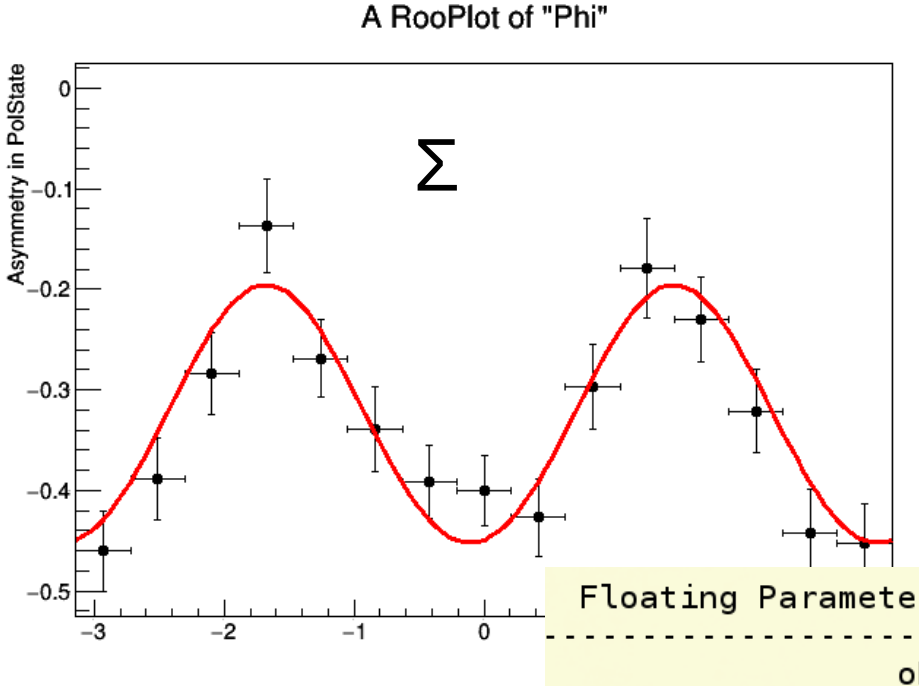
polarisation

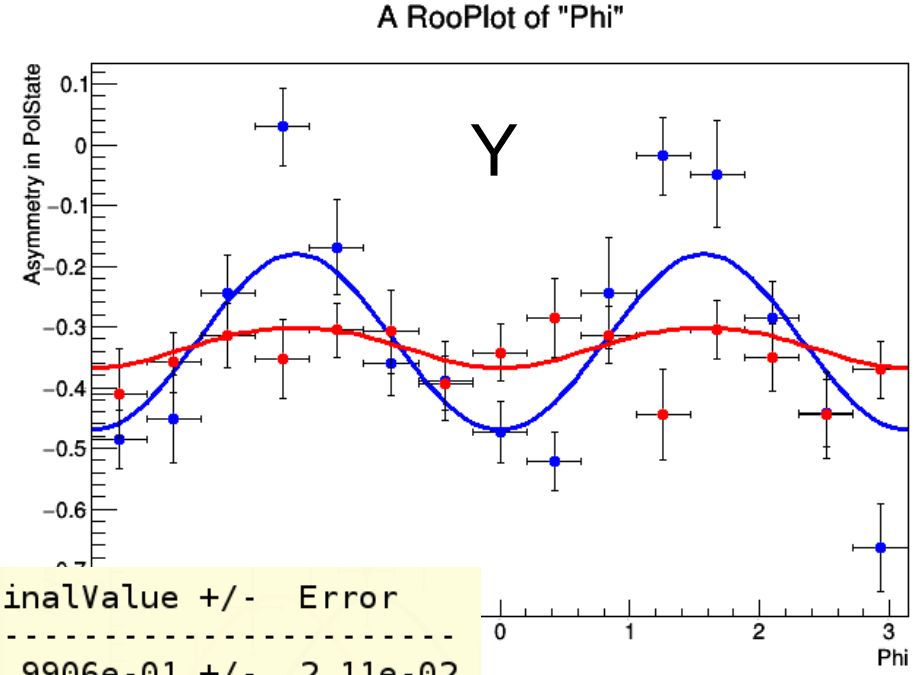
4 variables

5 free paramters

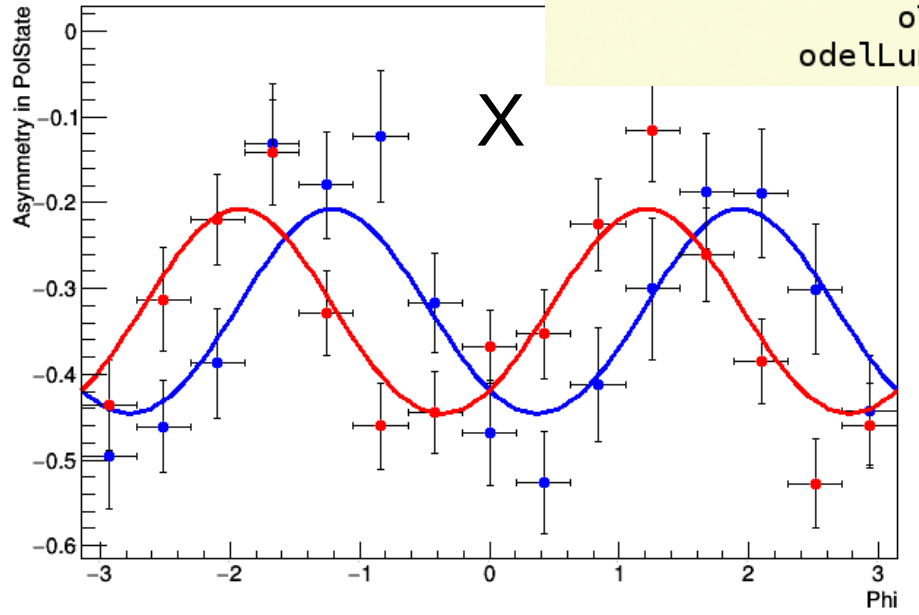
From previous sWeight fit

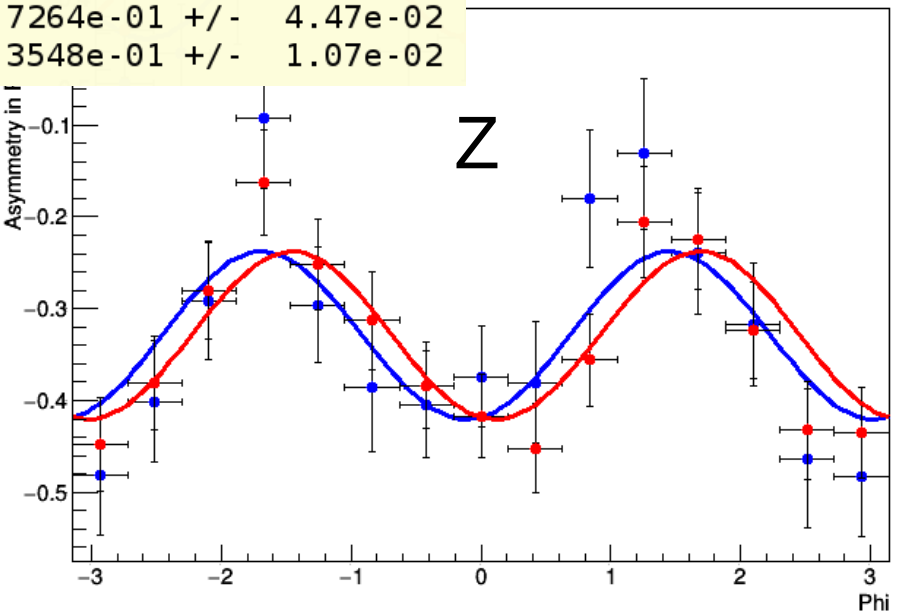
Azimuthal asymmetries with cuts on ± decay angles





Floating Parameter	FinalValue +/- Error
oB	-1.9906e-01 +/- 2.11e-02
o0x	5.3010e-01 +/- 5.89e-02
o0z	-1.3749e-01 +/- 6.40e-02
oT	-3.7264e-01 +/- 4.47e-02
odelLum	-3.3548e-01 +/- 1.07e-02





Plans (I)

Test methods for Ndim acceptance corrections
- Extended maximum likelihood
- Event-by-Event Weighting...

Implement Simulation validator for observable extraction (systematic uncertainties)

Fit with weights in AmpTools

Develop reliable amplitudes for fitting with

Add additional minimisers (other than Minuit)
- Genetic Algorithm
- Nested Sampling...

Kinematic Fitting, Q-factor, TMVA, …

Write an analysis note(s)

Plans (II)

Improve system on CLAS data, publish results

Test system as part of overall CLAS12 chain

Simulate benchmark channels