Contents

PSU software and spin information.

A few files define the relationship between the bunch number (Bunchid7bit) and yellow or blue spin.

Files are:

./root12fms/fmstxt/fill.txt - a list of runs with associated fill numbers

./root12fms/fmstxt/spinpat/ blue-buckets-polarization-fillpatterns.fillxxxxxx.dat
./root12fms/fmstxt/spinpat/ yell-buckets-polarization-fillpatterns.fillxxxxxx.dat
–cdev files for blue or yellow beams for fill xxxxx

Using Current Software to organize spin information.

To analyze trigger files, we create an instance of class **dataSet** (see **dataSet.h**)

Trigger files (1 or more) in the current directory will be accessed with calibration files pointed to by the instance of **FilesSet** called **p_files**.

We create an instance of **dataset** that represents one or more trigger data files (run12xxxxxxx.root)

dataSet d("./run*.root",p_files, hntName);

A **Fill** object is created within **d** (with pointer **d.RFill**), which contains an entry for each fill in a specified range of fill numbers that is implied by the run list (based on the text file **fill.txt**).

Also, a **TObjArray** is created with an entry for each run in the implied range. It is called

d.RunsArray

The runs from **fill.txt** that span the range indicated by "./run*.root" have corresponding entries in the array. The entries are instances of class **RunData**.

We step through the dataset **d** that contains **nevents = *d.input->GetEntries()*** events, reading each event "n" (***0<= n<nevents)*** with ***d.GetEntry(n);***

The pointer **d.thisRunDat** points to the current **RunData** entry from the array, which can change if we read an event that changes the run number of the set of trigger data files considered. If the **d.GetEntry(n)** changes from one trigger data file to another, then the variable **d.thisRunDat** is updated.

The run number is obtained from the name of **run*.root** file name that is currently being read.

Use of Fill class

The **Fill** class is created with

1) A pointer to a **FilesSet** structure (ie. **p_files**)
2) A first fill number
3) A last fill number
4) A first run number
5) A last run number

These ranges imply limits on the run/fill range that will be represented by an instance of this class (**Fill* Rfill)**

The association between run and fill is provided by the file "**fill.txt**".
The spin pattern for a particular run/fill is in the file "**spinpat/yell**....." or "**spinpat/blue**.....".

The most central methods of the **Fill** class are (instance **RFill**)

RFill->BlueSpin(bunchid7bit);
RFill->YellowSpin(bunchid7bit);

that return an integer 1,0, or -1 to indicate the polarization direction of the particuclar bunch.

Access to spin for an event.

Some members of the **dataSet** object "**d**" are:
d.RFill
d.bunchid7bit
d.CurrentRunNumber
d.CurrentSegNumber
d.BlueSpin
d.YellowSpin
d.kicked

Every time **d.GetEntry(n)** is called, the above variables are updated as necessary.

Making OFile.root files.

When **OFiles** are created, including a tree **p_out**, the variables
p_out->Bunchid7bit
p_out->spin
are created.

The **spin** variable contains blue and yellow spin up/down information
spin: {0,1,2,3}={bluedn_yellowdn, bluedn_yellowup, blueup_yellowdn, blueup_yellowup};
spin>3 for bad or missing bunch.

Making Output.root files.

We have the same definition as in **OFile.root** files for

TwoTr->spin
TwoTr->Bunchid7bit

Examples reading Output.root files.

The following two examples read an Output.root that contains events from four runs.
12095006
12095008
12095021
12095023

These runs all are part of the same fill.

Example I: Asymmetry (blue) for 1 photon events.

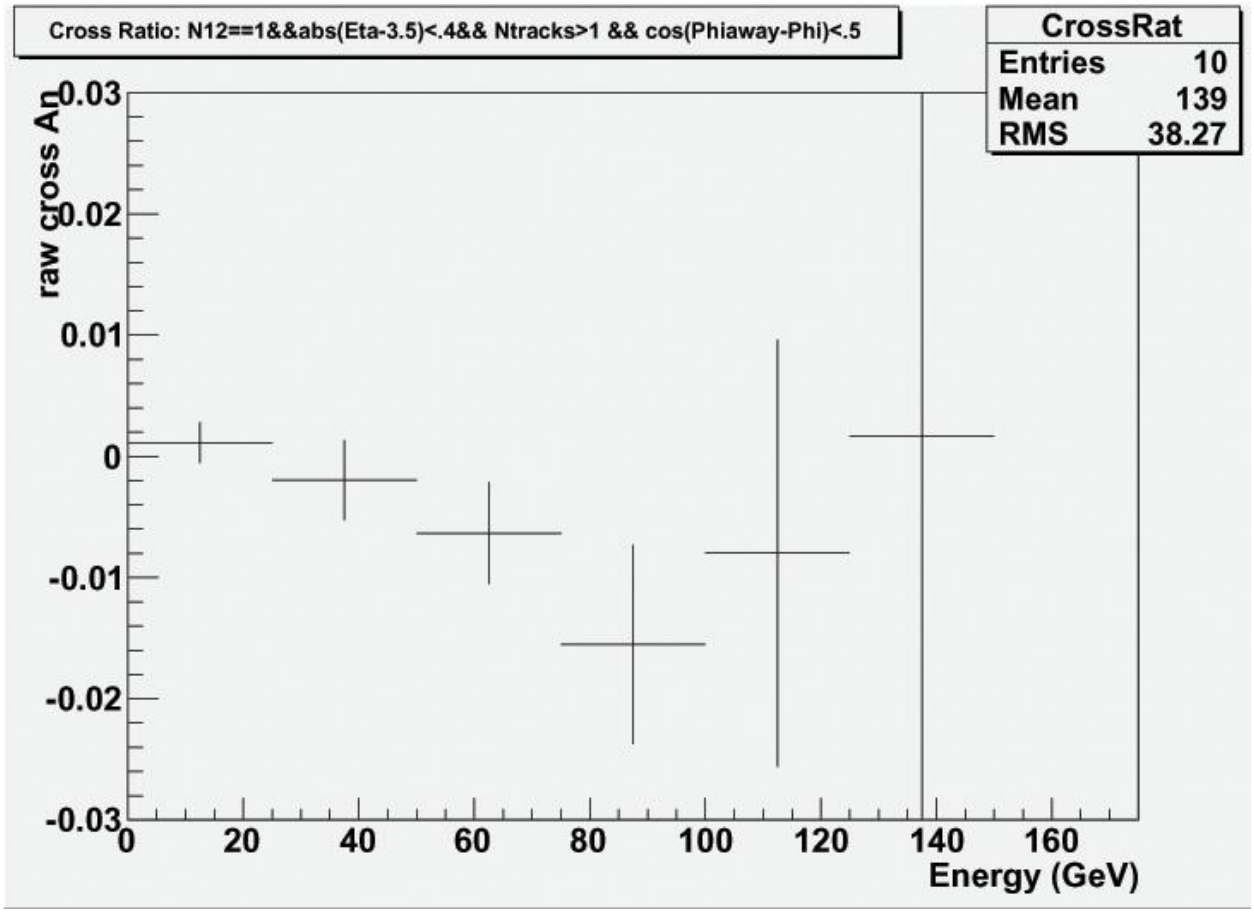
Table 1: Contents of Spin4.C.
Energy depencence of cross ratio for isolated 1 photon events.

```
{
  gROOT->Macro("start.C");
  AnalTools at;
  TCanvas* c2=new TCanvas("c2","c2",800,600);
  TCanvas* c1=new TCanvas("c1","c1",800,600);
  TFile f("Output95_4.root");
  TTree* TwoTr=f.Get("TwoTr");
  Int_t nbins=10 ;
  TH1F* E12uN=new TH1F("E12uN","uN",nbins,0,250.);
  TH1F* E12dN=new TH1F("E12dN","dN",nbins,0,250.);
  TH1F* E12uS=new TH1F("E12uS","uN",nbins,0,250.);
  TH1F* E12dS=new TH1F("E12dS","dN",nbins,0,250.);
  TString cuts1="N12==1&&abs(Eta-3.5)<.4&& Ntracks>1 && cos(Phiaway-Phi)<.5";

  TString cuts=cuts1;
  TString North="&&cos(Phi)<-.5";
  TString South="&&cos(Phi)>.5";
  TString Bdown="&&abs(spin-.5)<.6";
  TString Bup="&&abs(spin-1.5)<.6";
  TString Ydown="&&spin<4&&(spin%2==0)";
  TString Yup="&&spin<4&&(spin%2==1)";
  TString dn=Bdown;
  TString up=Bup;
  TwoTr->Project("E12uN","E12",cuts+North+dn);
  E12uN->SetLineColor(1);
  TwoTr->Project("E12dN","E12",cuts+North+up);
  E12dN->SetLineColor(2);
  E12uN->Draw("e");
  E12dN->Draw("esame");
  E12uN->Sumw2();

  E12dN->Sumw2();
  TH1F* RN=new TH1F((*E12uN)/(*E12dN));
  TwoTr->Project("E12uS","E12",cuts+South+dn);
  E12uS->SetLineColor(3);
  TwoTr->Project("E12dS","E12",cuts+South+up);
  E12dS->SetLineColor(4);
  E12uS->Draw("esame");
  E12dS->Draw("esame");
  E12uS->Sumw2();
  E12dS->Sumw2();
  c2->cd();
  TH1F* RS=new TH1F((*E12uS)/(*E12dS));
  RN->Draw("e");
  RS->SetLineColor(4);
  RS->Draw("esame");
  TH1F* crossR=new TH1F(at.GetCross(E12uN,E12dN,E12uS,E12dS,"CrossRat"));
  TCanvas* c3=new TCanvas("c3","c3",800,600);
  TString ctit="Cross Ratio: ";
  ctit=ctit+cuts;
  crossR->SetTitle(ctit);
  crossR->GetXaxis()->SetTitle("Energy (GeV)");
  crossR->GetYaxis()->SetTitle("raw cross An");
  crossR->Draw();
  crossR->GetXaxis()->SetRangeUser(0,150);
  crossR->GetYaxis()->SetRangeUser(-.03,.03);
  c3->Print("cr_E_1ph.ps");
  system("ps2pdf cr_E_1ph.ps");

}
```


Cross Ratio: N12==1&&abs(Eta-3.5)<.4&& Ntracks>1 && cos(Phiaway-Phi)<.5
CrossRat
Entries 10
Mean 139
RMS 38.27
raw cross An
0.03
0.02
0.01
0
-0.01
-0.02
-0.03
0
20
40
60
80
100
120
140
160
Energy (GeV)

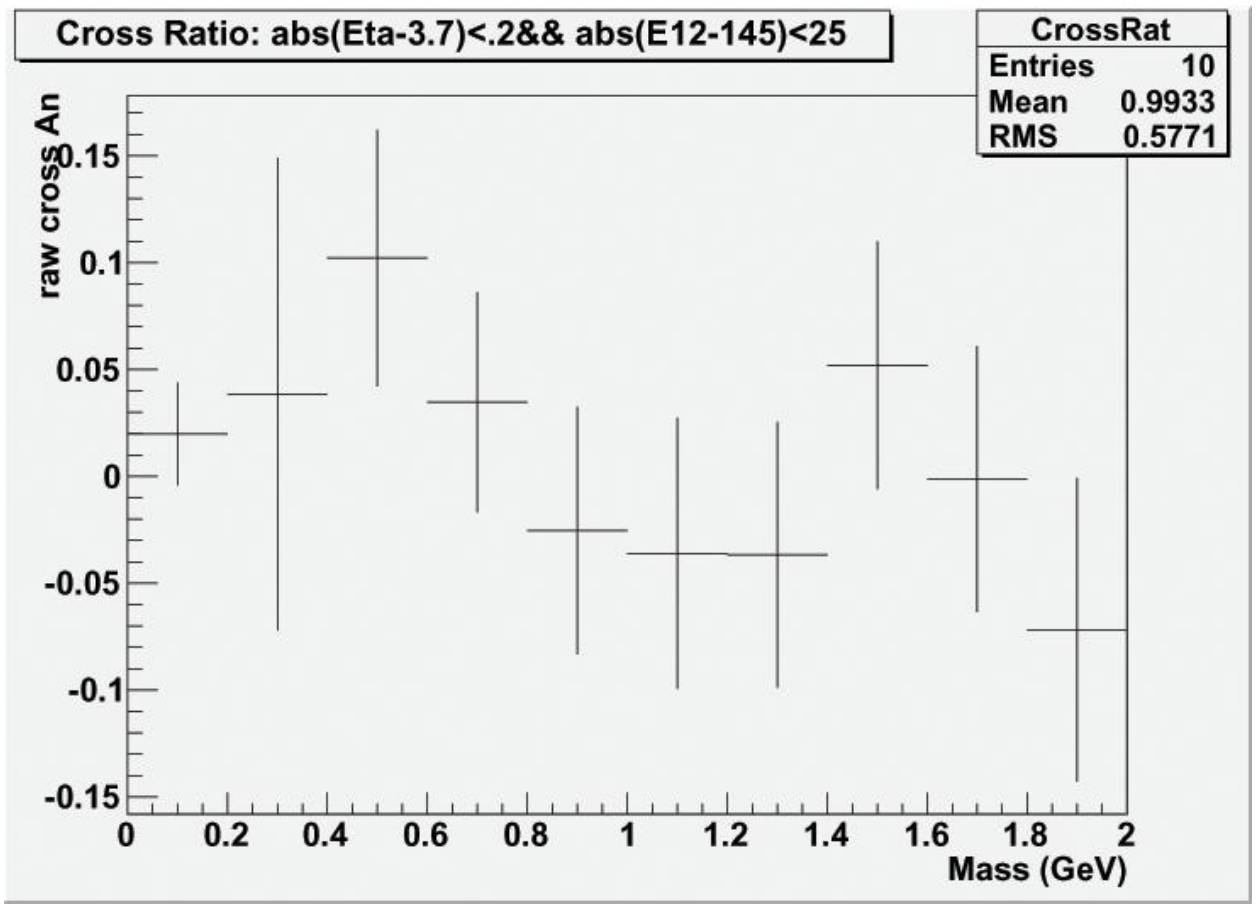
Example II: Asymmetry (blue) in Eta region, 120GeV < E<170 GeV.

Table 2: Contents of Spin3.C.
Mass dependence of cross ratio asymmetry.

```
{
  gROOT->Macro("start.C");
  AnalTools at;
  TCanvas* c2=new TCanvas("c2","c2",800,600);
  TCanvas* c1=new TCanvas("c1","c1",800,600);
  TFile f("Output95_4.root");
  TTree* TwoTr=f.Get("TwoTr");
  Int_t nbins=10 ;
  TH1F* M12uN=new TH1F("M12uN","uN",nbins,0,2.);
  TH1F* M12dN=new TH1F("M12dN","dN",nbins,0,2.);
  TH1F* M12uS=new TH1F("M12uS","uN",nbins,0,2.);
  TH1F* M12dS=new TH1F("M12dS","dN",nbins,0,2.);
  TString cuts1="abs(Eta-3.7)<.3&& abs(E12-40)<20&&N12==2";
  TString cuts1="abs(Eta-3.7)<.2&& abs(E12-145)<25";

  TString cuts=cuts1;
  TString North="&&cos(Phi)<-.5";
  TString South="&&cos(Phi)>.5";
  TString Bdown="&&abs(spin-.5)<.6";
  TString Bup="&&abs(spin-1.5)<.6";
  TString Ydown="&&spin<4&&(spin%2==0)";
  TString Yup="&&spin<4&&(spin%2==1)";
  TString dn=Bdown;
  TString up=Bup;
  TwoTr->Project("M12uN","M12",cuts+North+dn);
  M12uN->SetLineColor(1);
  TwoTr->Project("M12dN","M12",cuts+North+up);
  M12dN->SetLineColor(2);
  M12uN->Draw("e");
  M12dN->Draw("esame");
  M12uN->Sumw2();

  M12dN->Sumw2();
  TH1F* RN=new TH1F((*M12uN)/(*M12dN));
  TwoTr->Project("M12uS","M12",cuts+South+dn);
  M12uS->SetLineColor(3);
  TwoTr->Project("M12dS","M12",cuts+South+up);
  M12dS->SetLineColor(4);
  M12uS->Draw("esame");
  M12dS->Draw("esame");
  M12uS->Sumw2();
  M12dS->Sumw2();
  c2->cd();
  TH1F* RS=new TH1F((*M12uS)/(*M12dS));
  RN->Draw("e");
  RS->SetLineColor(4);
  RS->Draw("esame");
  TH1F* crossR=new
  TH1F(at.GetCross(M12uN,M12dN,M12uS,M12dS,"CrossRat"));
  TCanvas* c3=new TCanvas("c3","c3",800,600);
  TString ctit="Cross Ratio: ";
  ctit=ctit+cuts;
  crossR->SetTitle(ctit);
  crossR->GetXaxis()->SetTitle("Mass (GeV)");
  crossR->GetYaxis()->SetTitle("raw cross An");
  crossR->Draw();
  c3->Print("cr_M_E145.ps");
  system("ps2pdf cr_M_E145.ps");
}
```


Cross Ratio: abs(Eta-3.7)<.2&& abs(E12-145)<25
CrossRat
Entries 10
Mean 0.9933
RMS 0.5771
raw cross An
0.15
0.1
0.05
0
-0.05
-0.1
-0.15
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
2
Mass (GeV)