MUON ARMS TUTORIAL

for J/ψ Analysis

Contents

1 Introduction

This tutorial is a brief introduction to J/ψ analysis in the PHENIX Muon Arms. For this tutorial, we will point you to a set of pDST files to use if you do not yet have them available. There are about a half dozen steps taken in order to extract the J/ψ signal, and includes running on CONDOR twice. Here we will work through an example to extract the J/ψ yield as a function of p_T in the Run15pAu data set for all tracks detected in the Muon Arms.

The steps outlined in this tutorial are the following:

Analysis Chain for Real Data

1. Introduce the pDST analysis code
2. Run over the pDST files with CONDOR to create histograms for your selected tracks and variables
3. Use the built-in command **hadd** to combine the CONDOR output files into a single root file
4. Run the **Sigmalizer** macro to optimize Muon Arm track cuts
5. Run over pDST files with CONDOR again using the updated track cuts
6. Use a rebinning macro to take the output histograms and create different p_T binning
7. Fit the rebinned mass histograms and look for indications of stable fits

2 CVS

In CVS, there is a directory which includes all the macros discussed in this tutorial that can be checked out to your local working area on RACF using the command:

cvs co offline/analysis/JpsiAnalysis_MuonArms

If the command **cvs co** does not work, the following can also be used:

setenv CVSROOT /afs/rhic/phenix/PHENIX_CVS
cvs checkout offline/AnalysisTrain/JpsiAnalysis_MuonArms

If you receive a "Permission denied" message when trying to use CVS, try obtaining an AFS token first using this command:

/usr/bin/klog.krb5

and then enter your password at the prompt. More information can be found here regarding this command: BNL web page.

A back-up directory **JpsiAnalysis_MuonArms.tar** has been added to HPSS. A small number of pDST files have also been archived under the directory name **JpsiAnalysis_MuonArms_pDST.tar** To access these files from HPSS, the interface utility **HSI** can be used by entering the command:

hsi

At the prompt to enter the "Kerberos Principal", enter your Kerberos (RACF) username. At the next prompt, enter your Kerberos password. To copy the **.tar** file from HPSS to your current working directory, use the following command:

get /home/klsmith/JpsiAnalysis_MuonArms.tar

and wait for the file transfer to complete. Then close the **HSI** utility using:

exit

The **.tar** file can then be opened with:

tar -xvf JpsiAnalysis_MuonArms.tar

3 Preliminaries

3.1 J/ψ Invariant Mass

This analysis involves the reconstruction of the J/ψ invariant mass through the dimuon decay channel. This reconstruction process is handled entirely by the Analysis Module **picoDST object** when you "Ride the Taxi". To reconstruct the J/ψ invariant mass, the momenta of single muons is measured by the Muon Arms, and the mass of each muon is assigned (all particles that fire in the MuID are recorded as muons). Conservation of four-momentum is used to determine the J/ψ invariant mass:

$$q_1 = q_2 + q_3$$

$$q_1^2 = (q_2 + q_3)^2$$

$$q_1^2 = m_\mu^2 + m_\mu^2 + 2E_2E_3 - 2\bar{p}_2 \cdot \bar{p}_3$$

$$q_1^2 = (m_{\mu\mu}^{J/\psi})^2$$

The picoDST object calculates the J/ψ invariant mass using the above formula, and results in a distribution like the one shown below.

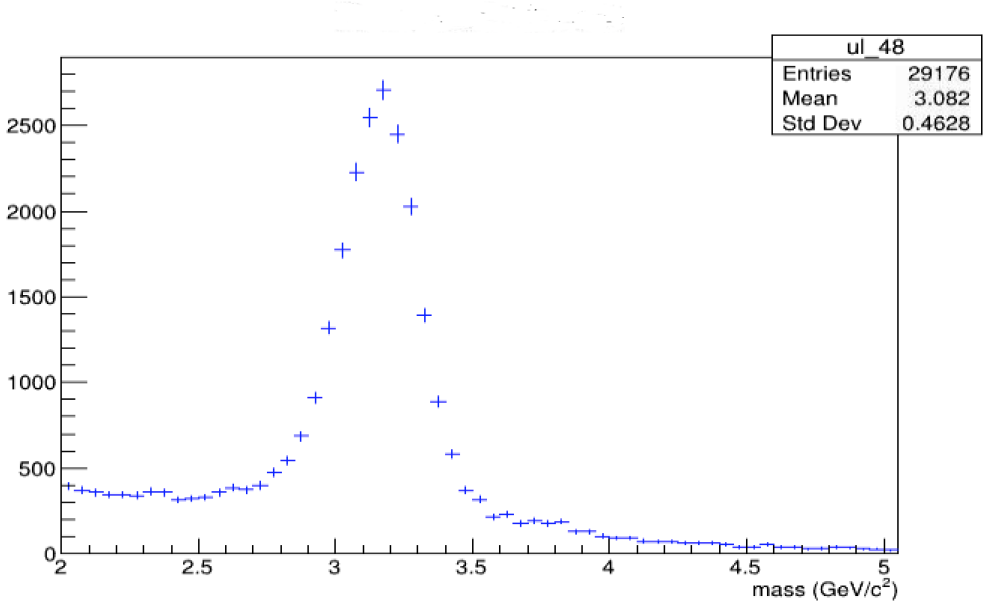

Figure 1: Example J/ψ invariant mass distribution in Run15pAu North

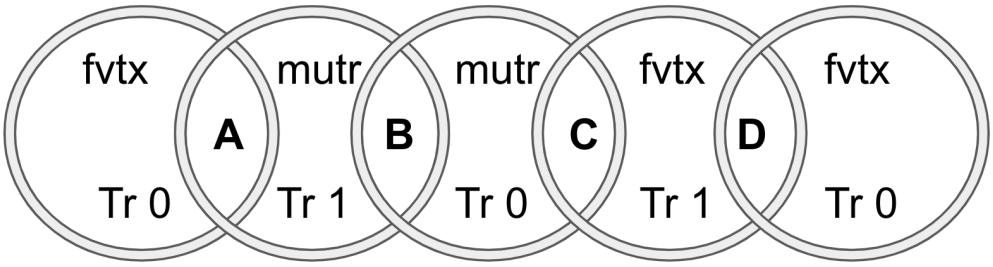

Figure 2: All possible dimuon track combinations in the Muon Arms

3.2 Muon Arm Tracks

The Muon Arms includes the **FVTX** and the **MuTr** detectors, and single muon tracks can be combined together in four different combinations, as shown above in the diagram. A single muon track in the **FVTX** detector is labeled "fvtx" and a single muon track in the **MuTr** detector is labeled "mutr". All four track combinations used together (Intersections A+B+C+D) represents the largest statistical set available in the Muon Arms. We have called this set "All Tracks", and will be using this set to work through the analysis example.

Another set used in Muon Arm analysis is "Fvtx tracks" (Intersection D) which is the case when both single muon tracks are detected by the **FVTX**. This is a smaller fraction of total tracks due to the geometric acceptance of the FVTX detectors (see figure below).

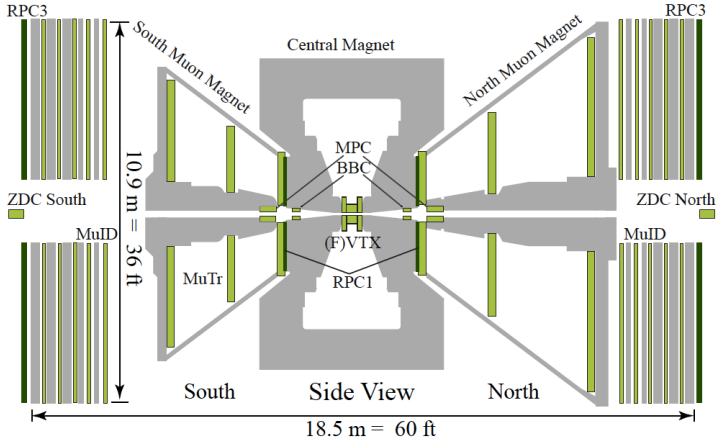

Figure 3: Side view of the PHENIX detector after installation of the FVTX (2012).

The advantage to using this set of tracks is the high resolution of the FVTX detector. The J/ψ signal in the Muon Arms is prominent above the background, and while the FVTX resolution can be used, it is not necessary for an accurate measurement. But for the measurement of the much weaker ψ(2S) signal for example, the MuTr resolution alone is not enough.

3.3 DiMuon Container

The DiMuon Container should be one of the nodes on the TTree contained in your pDST files for J/ψ analysis after using the PHENIX Taxi. The DiMuon Container is part of the larger **picoDST object** analysis module, which is located in CVS here:

offline/AnalysisTrain/picoDST_object

The module was created and is maintained by the FVTX group at Los Alamos. Information about the variables in the DiMuon Container and the data type of each variable can be viewed at this PHENIX web page: **Software Page**

The DiMuonContainer variables hold essentially everything you need to extract J/ψ mass histograms from the pDST files. For an analysis involving the set of all Muon Arm tracks, the reconstructed invariant mass is stored in the variable **mass** and the kinematic information in the variables **px, py, pz**, etc. For an analysis involving the Fvtx tracks, the reconstructed invariant mass is stored in the variable **mass_fvtxmutr** and the associated kinematic information in the variables **px_fvtmutr, py_fvtxmutr, pz_fvtxmutr**, etc.

Accessing the variables in a ROOT TTree can be done using the **TTreeReader**. The TTreeReader is not available in the PHENIX framework (ROOT 5), but is available in the sPHENIX framework (ROOT 6). Chris Pinkenburg provided the following command for access to sPHENIX ROOT 6:

source /opt/sphenix/core/bin/sphenix_setup.csh -n

The TTreeReader needs the name of the variable and the node in the TTree, the data type, and whether the variable is single (TTreeReaderValue) or multi valued (TTreeReaderArray):

```
TTreeReaderArray<float> Tr0_rap(myReader, "DiMuons.Tr0_rapidity");
TTreeReaderArray<float> Tr1_rap(myReader, "DiMuons.Tr1_rapidity");
TTreeReaderArray<float> px_Tr0(myReader, "DiMuons.Tr0_px");
TTreeReaderArray<float> px_Tr1(myReader, "DiMuons.Tr1_px");
TTreeReaderArray<float> py_Tr0(myReader, "DiMuons.Tr0_py");
TTreeReaderArray<float> py_Tr1(myReader, "DiMuons.Tr1_py");
TTreeReaderArray<float> pz_Tr0(myReader, "DiMuons.Tr0_pz");
TTreeReaderArray<float> pz_Tr1(myReader, "DiMuons.Tr1_pz");
```

Figure 4: TTreeReader format for several multi value DiMuon Container variables.

More examples of DiMuon Container variables in the correct TTreeReader format can be found around lines 200 - 250 in the analysis code:

Run15pAu_TTree_NOFVTX.C

Note that you will need to source the ROOT 6 sPHENIX setup to run this code locally in your area and also when submitting jobs to CONDOR that will use the TTreeReader.

3.4 Total Fit Components

There are four main components in the fit to the J/ψ invariant mass distribution: [1]**the combinatorial background fit**, [2]**the correlated background fit**, [3]**the signal fits** and [4]**the total fit**, which is simply the sum of all fits. Different functions can be used to fit the different components of the distribution. For this tutorial, we have used the modified Hagedorn function for the combinatorial background, a simple exponential function for the correlated background, and a Crystal Ball function for the J/ψ and ψ(2S) signals. Here we define these functions below.

Modified Hagedorn function:

$$f(m) = \frac{p_0}{[\ \exp\left(-p_1 m - p_2 m^2\right) + m/p_3\ ]^{p_4}} \tag{1}$$

Simple exponential function:

$$f(m) = p_0\ exp(\ p_1 m\ ) \tag{2}$$

Crystal Ball function:

$$f(m) = N \cdot \exp\left(-\frac{(m-\bar{m})^2}{2\sigma^2}\right), \ \text{ for } \ \frac{m-\bar{m}}{\sigma} > -\alpha \tag{3}$$

$$f(m) = N \cdot A \cdot \left(B - \frac{(m-\bar{m})^2}{\sigma}\right)^{-n}, \ \text{ for } \ \frac{m-\bar{m}}{\sigma} \leq -\alpha$$

$$A = \left(\frac{n}{|\alpha|}\right)^n \cdot \exp\left(-\frac{|\alpha|^2}{2}\right), \ B = \frac{n}{|\alpha|} - |\alpha|$$

The combinatorial background consists of "muon" pairs which are not kinematically related (a fraction of muons are misidentified hadrons). Either the "like-sign" background or the "mixed events" background can be used, and both options are included in this tutorial. The "like-sign" background is collected by selecting muons from the same event that have the same charge, while mixed events are muons of opposite charge from different events.

The correlated background consists of "muon" pairs that are kinematically related. There are several different contributors to the correlated background, including open heavy flavor decays, D-meson decays and Drell-Yan processes.

The total fit is typically evaluated using χ^2/NDF, or the chisquare per number of degrees of freedom. In general, the closer the ratio is to 1, the better the fit. However, keep in mind that a good χ^2/NDF does not guarantee the fit is reliable. Also keep in mind that there are a range of fits possible for each distribution, and the first fit that converges may not be the best choice.

3.5 Fit Stability and Cross Checks

Fitting a mass distribution is not foolproof, and should be done with care. There are usually four status messages provided by ROOT after the evaluation of a fit:

CONVERGED, NOT POSDEF, OK, FAILED

• Never consider the result from a **FAILED** fit status. A fit status of **CONVERGED** is the safest option, but **POSDEF** and **OK** status can be used

• A first derivative is listed for each parameter in the fit result, and should not be equal to zero

• The correlation matrix that prints to the screen should not have entries that are zero

• The statistical error for a fit must always be larger than the square root of the J/ψ yield. It is a good idea to have an **if statement** in your macro that prints a message whether or not the statistical error is indeed larger than the square root of the yield

• It is good practice to sum yields over a variable that has been divided into finer bins and check if the sum is consistent with the yield from a single fit result taken over the entire range. In this tutorial you can check that the sum of yields over the 24 p_T bins taken from 0-6 GeV/c is consistent with the single yield fit over the range 0-6 GeV/c

• You might also be interested in checking if fits using different functions return yields that are similar. For example, fitting the correlated background with the modified Hagedorn function as well as the simple exponential function

• Comparing the yield results between the like-sign background and the mixed events background is also a good check (see figures on next page)

• A final possibility is to take the sum of the correlated background and the combinatorial background and check that it follows the same base curve as the total fit function across the full mass range (see figures on next page).

• For cases when statistics are low, the chances for fit stability decrease. It is helpful to fix the J/ψ mass to the PDG reported value, or to the p_T integrated fit value, and include a systematic uncertainty with the measurement

• It could also be useful to fix the J/ψ width. A systematic would also need to be included

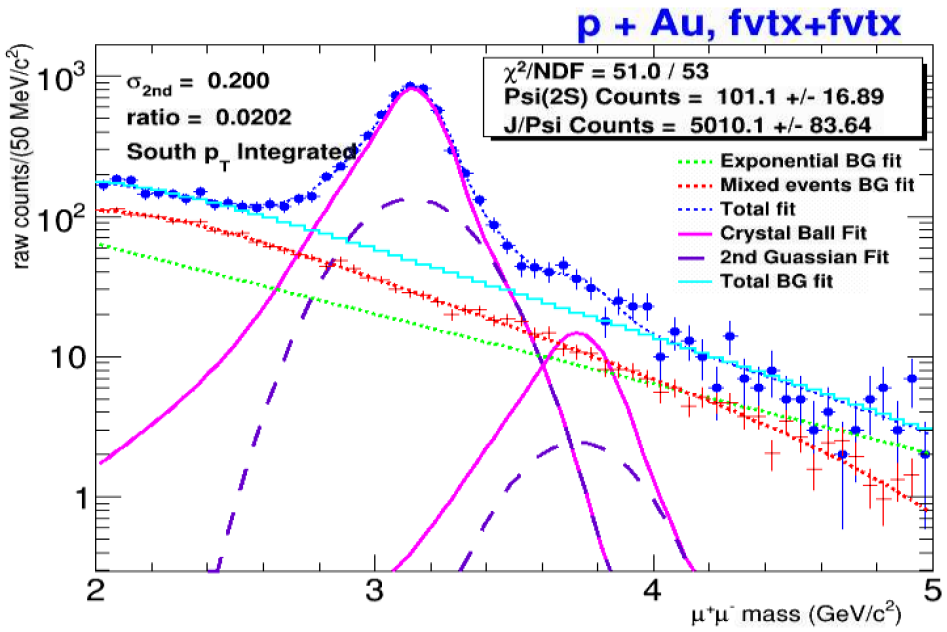

Figure 5: Example fit using "Ftvx tracks". This measurement is for the $\psi(2S)$, and uses an additional fit under the Crystal Ball curve (a Guassian function). The light blue fit represents the sum of the correlated and combinatorial background, and indeed follows the curve of the total fit function under the peaks.

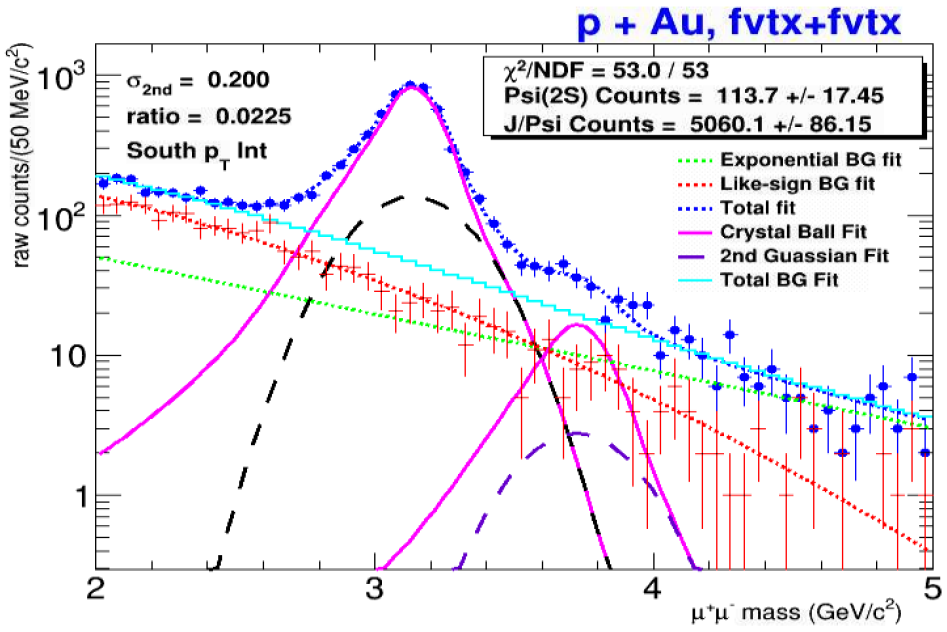

Figure 6: This is the same fit shown above except the mixed events background has been replaced with the like-sign background as a cross-check. There is about a 1% difference in the J/ψ yield, and therefore we can conclude the different background methods give consistent results.

4 Analysis Chain Steps

4.1 Pre-Step: pDST Files

For this tutorial, we will point you to a set of Run15pAu pDST files available in Matt Durham's area that can be used for this exercise.

4.2 Step 1: pDST Analysis Code

We have included an example of a Muon Arms pDST analysis code provided by Matt Durham for the "All Tracks" Muon Arms statistics set:

Run15pAu_TTree_NOFVTX.C

The general structure of the macro is to first read in the text files with the pathnames to the pDST directory. Then histograms are defined that will be filled with selected variables from the pDST TTree nodes. To access the variables in the TTree nodes, the macro uses the **ROOT TTreeReader**. Once the pDST files are located, the histograms have been created and the variables in the TTree accessed, then we can use a for loop to run over all the events contained in the run and fill the histograms with the selected variables and track cuts. After the histograms have been filled, they are written out to a root file.

There are a few additional details to note. If the variable is single valued, such as the run number, the correct format for the TTreeReader is **TTreeReaderValue**. If the variable is multi-valued, as most variables in the TTree, the correct format is **TTreeReaderArray**. Also, the macros used in Steps 6 and 7 are formatted to read in the histograms defined in this analysis macro with the exact same names, total bin numbers and axis ranges.

For this exercise, there are not any changes that need to be made to the macro, and it is ready to be submitted to CONDOR. But in practice you will want to test your code before submitting it. To test the code locally, there are a few things to change:

1. Comment out: line 14 (arguments for the macro), line 38 (reads in text file)
2. Uncomment: line 16 (no arguments for the macro), line 19 (an input pDST file), line 34 (reads in root file) and line 626 (define an output file)

Then the macro can be test run with **root -l Run15pAu_TTree_NOFVTX.C**. If there are no errors and the output root file has histogram entries that you would expect for a single physics run, then change the macro back to the CONDOR settings before submitting.

4.3 Step 2: Analyze pDSTs with CONDOR

Codes from the **JpsiAnalysis_MuonArms** directory (see section 2) used in this step:

CONDOR/jobfiles/Lines.bash
CONDOR/analyze_pAu_NOFVTX.condor
CONDOR/analyze_pAu_NOFVTX.csh
Run15pAu_TTree_NOFVTX.C

To start, first cd into the following area to create a temporary directory for the CONDOR log files (recommended method by PHENIX):

cd /home/tmp/
mkdir < username >

and name the directory with your RACF username. Then source the ROOT 6 setup (discussed in section 3.3) using the command:

source /opt/sphenix/core/bin/sphenix_setup.csh -n

Then cd back into your previous directory using the following command: **cd -**

From here cd into the jobfiles directory:

cd offline/analysis/JpsiAnalysis_MuonArms/CONDOR/jobfiles

and make a text file listing the pathnames to the directory containing the pDST files. For this tutorial, we will use the files located in Matt Durham's hhj3 area:

/phenix/hhj3/durham/taxi/Run15pAu105plus108/

To generate the list, use the following command:

ls /phenix/hhj3/durham/taxi/Run15pAu105plus108/*.root > list.txt

Then open the text file and check there are 418 lines listed. CONDOR will analyze one pDST at a time, so we need to break the list into single pDST pathnames. This can be done using:

bash Lines.bash

which generates 418 individual text files inside the jobfiles directory. Then cd up one directory to CONDOR, and copy the full pathname that displays after typing the command **pwd**. The pathname can be highlighted and then copied from the drop down Edit menu in the terminal window or using

the keyboard command (control+command+c).

Open the **.condor** file (see figure on next page) and make the following changes:

1. Replace the pathnames for **Arguments, Initialdir, Output and Error**
2. Replace the username for the **Log** files
3. Replace the email for **Notify_user** with your email address

Next open the **.csh** file (see figure on next page) and make a similar change:

1. Replace the pathnames for **infile and outfile**

Then after these changes are made, you are ready to submit your job to CONDOR using the command:

condor_submit analyze_pAu_NOFVTX.condor

Jobs can run and complete within minutes or hours. To display the number of root files in a given directory, cd into the jobfiles directory and use the command:

ls *.root | wc -l

You can also check the number of jobs remaining on CONDOR using:

condor_q < username >

Note this should be done using the same machine you submitted from (rcas2063, rcas2072, etc)

```
Universe       = vanilla
Notification   = Never
Executable     = /bin/csh
Arguments      = /phenix/hhj/klsmith/Jpsis/Run15pAu/fit_NOFVTX/krista_fits/matt_centrality_macros/CONDOR/analyze_pAu_NOFVTX.csh $(Process)
Requirements   = CPU_Speed >= 1
Rank           = CPU_Speed
Image_Size     = 42000
Priority       = +20
GetEnv         = True
Initialdir     = /phenix/hhj/klsmith/Jpsis/Run15pAu/fit_NOFVTX/krista_fits/matt_centrality_macros/CONDOR
Input          = /dev/null
Output         = /phenix/hhj/klsmith/Jpsis/Run15pAu/fit_NOFVTX/krista_fits/matt_centrality_macros/output/Run15pAu_$(Process).out
Error          = /phenix/hhj/klsmith/Jpsis/Run15pAu/fit_NOFVTX/krista_fits/matt_centrality_macros/output/Run15pAu_$(Process).err
Log            = /home/tmp/klsmith/Run15pAu_$(Process).log
Notify_user    = kls15k@my.fsu.edu
Queue 418
```

Figure 7: Example condor file

```
#! /bin/csh


set tail = $1

echo $1
echo $tail

set infile =  "/phenix/hhj/klsmith/Jpsis/Run15pAu/fit_NOFVTX/krista_fits/matt_centrality_macros/CONDOR/jobfiles_tutorial/Run15pAu_NOFVTX_$tail.txt"

echo $infile

set outfile = "/phenix/hhj/klsmith/Jpsis/Run15pAu/fit_NOFVTX/krista_fits/matt_centrality_macros/CONDOR/jobfile_tutorials/Run15pAu_NOFVTX$tail.root"

echo $outfile

root -b -q ../Run15pAu_TTree_NOFVTX.C\(\"$infile\",\"$outfile\"\)
```

Figure 8: Example csh file

4.4 Step 3: Combine Output Files

The convenient and simple built-in command **hadd** can be used to add root files together containing histograms with exactly the same binning and range. If we name the new root file **result.root**, then the command would be the following:

hadd result.root Run15pAu_NOFVTX_*.root

4.5 Step 4: Optimize Track Cuts

Codes from the **JpsiAnalysis_MuonArms** directory used in this step:

sigmalizer_macro_durham.C

This code was also provided by Matt Durham, and optimizes the cuts for the track variables **DG0, DDG0, dr, dtheta, and dphi**. For the "All tracks" data set used in this example, we will not make any **dr, dtheta or dphi** cuts, since these are quality cuts associated with the "Fvtx" tracks.

Sigmalization is the process of taking a TH2 histogram and doing the following steps: projecting bins along the x-axis onto the y-axis, fitting the data points in each projection with a Gaussian, plotting the mean of the Gaussian fits as a function of x, fitting the distribution of the Gaussian means with the prescribed function f(p) (see Equation below), and then shifting the function f(p) by 5 standard deviations. Any data points outside of the 5 sigma curve are then cut. These series of steps are carried out by a single ROOT Method: FitSlicesY(f,xbin_min, xbin_max). The options f, xbin _min and xbin_max correspond to the function used to fit the projections, and the range of bins over the x-axis to project. '0' can be used for a Gaussian fit.

ROOT saves the histograms of each parameter as a function of x in your local directory, and labels the histograms using the same name with the labels '_0' for the Gaussian normalization parameter, '_1' for the Gaussian mean and '_2' for the Gaussian sigma. The sigmalizer code then uses the 'gDirectory' to retrieve the histograms from your directory.

The function used to fit the distribution of Gaussian means is the following:

$$f(p) = a + \frac{b}{\bar{p}^2}, \tag{4}$$

where a and b are parameters and $\bar{p}$ is the total momentum of each track. To use the sigmalizer macro:

1. Replace the name of the TFile at the top with **result.root**, or the name of the file generated in the previous step
2. Run the macro using **root -l sigmalizer_macro_durham.C** and copy the results that print to the screen
3. Then replace lines 439 - 448 in the analysis macro **Run15pAu_TTree_NOFVTX.C** with the new signalization cuts

4.6 Step 5: Re-Analyze pDSTs with CONDOR

This step is very straight forward after setting up the directories and scripts to submit jobs in Step 3. After replacing the sigmalizer cuts in Step 4 and saving them, cd into the CONDOR directory and resubmit the jobs:

condor_submit analyze_pAu_NOFVTX.condor

All previous root files will be overwritten. After CONDOR completes, again merge the files together using **hadd**

4.7 Step 6: Rebin Histograms

Codes from the **JpsiAnalysis_MuonArms** directory used in this step:

rebinning_macro_pt_version.C

Having macros that can read in histograms and rebin them can be useful when CONDOR is busy. The example macro above can rebin histograms in various p_T binwidths. For this tutorial, we will use binwidths of 0.25 GeV/c, with a p_T initial value of zero. The macro has options for 0.50 GeV/c binwidths as well as 1.0 and 2.0 GeV/c binwidths, which are all commonly used in J/ψ analysis. The initial p_T value can also be changed.

To use the rebinning macro:

1. Replace the name of the TFile at lines 62 and 74 with your new pDST root file from Step 5
2. Run the macro using **root -l rebinning_macro_pt_version.C** with the North arm (at line 41) set to true
3. Repeat for the North arm set to false

This will generate a North and South root file with histogram names and p_T binning that the fitting macro in Step 6 is already formatted to read in.

The p_T integrated fit result is located in bin 49, which is the **ROOT Projection** that takes place outside of the **for loop** at lines 148, 172, 196, and 219. Currently the p_T integrated bin runs from 0-12 GeV/c, which corresponds to bins 1-48 for 0.25 GeV/c binwidths. The p_T integrated range can easily be changed with no need to change the array element (currently set to 48). For example, projecting bins 1-24 would correspond to a p_T integrated bin of 0-6 GeV/c.

4.8 Step 7: Fit Mass Spectra

Codes from the **JpsiAnalysis_MuonArms** directory used in this step:

"CBFitter_exp_bg.C"

The Crystal Ball fitting macro is formatted for p_T dependent measurements. The macro fits one p_T bin at a time, which can be selected using the bin number at the prompt. You can run the macro using **root -l CBFitter_exp_bg.C+**. A few example plots of various p_T bins are shown on the next page for reference. The plots were produced using this same fitting macro, and if all the steps were followed correctly, you should expect to see similar results.

Outlining the general structure of the macro, it begins with the function definitions at the top. The fitting code starts around line 164, and just below that is a bool variable for the North and South arms. The prompts that print to the screen ask for the bin number, whether or not to use a combinatorial background (as you go to higher p_T bins, the background hits become less prominent), and which set of combinatorial fit parameters to use. The last prompt is referring to the lines in the code between 290 and 413, which are the parameter initializations for the modified Hagedorn fit to the combinatorial background. If you run the macro using the compiler (using a +), the combinatorial background parameters can be cycled through to look at a range of different fits. The option to change from like-sign to mixed events background can be found around line 424 and again at line 746. Using the command "control+s" you can search the code for other instances that need to be changed.

Following the combinatorial background fit is the total fit function, which has 10 parameters when using an exponential function. These are initialized around line 460. Note that if you change the exponential function (2 parameters) to the Hagedorn function (5 parameters), you will need to change the number of parameters in the total fit function defined around line 106. This change will propagate through the macro in various places. In particular, the correlation matrix for the ψ(2S) will need to be modified (lines 595 - 627), which calculates the error in the yield.

After all functions are defined, the data file is read in, fit parameters are initialized, the fits are applied to the data, and the correlation matrices are calculated, the fits and histogram components are then drawn to the same plot. Various fit information can be added to the plot, such as p_T range, rapidity, system, track type, and most importantly, **the extracted J/ψ yield and statistical uncertainty**.

Lastly, to perform the cross check suggested in section 3.5, run the rebinning macro again as described in section 4.7. The p_T integrated bin corresponds to **bin number 49** and can be modified to run from 0-6 GeV/c. The extracted yield from fitting this bin should be within a percent or two from the sum of yields taken over all 24 p_T bins.

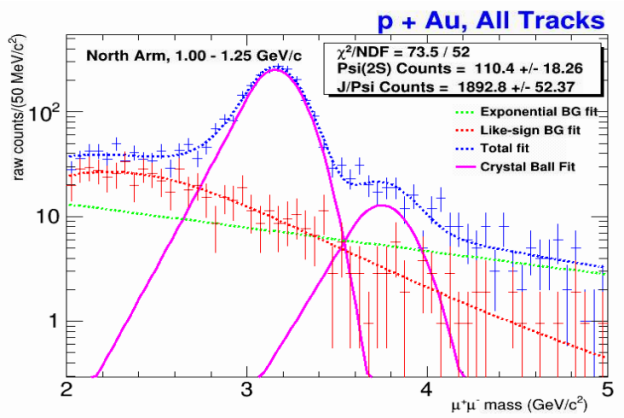

Figure 9: Example fit in North arm for 1.00-1.25 GeV/c (bin 5).

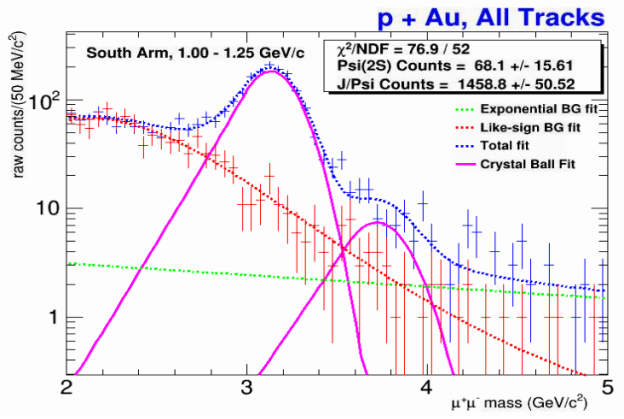

Figure 10: Example fit in South arm for the same p_T range.

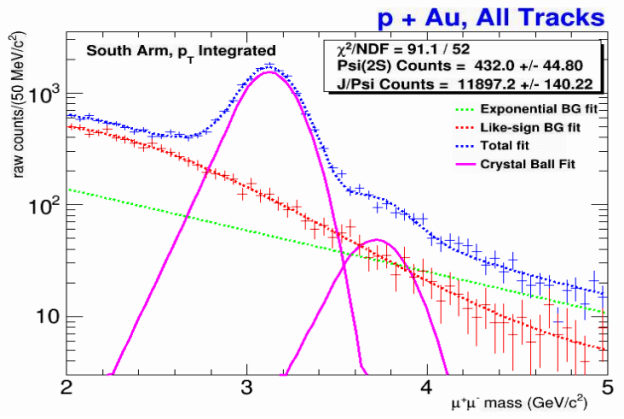

Figure 11: Example p_T integrated fit for 0-6 GeV/c in the South arm (bin 49).