PID-by-dE/dx guide

UCD Nuclear Physics Group

February 12, 2003

Abstract

This is a guide to using the software developed by Juan Romero. The software allows us to obtain particle dE/dx distributions, specify cuts, change calibration parameters, and create Gaussian fits to identify particles so we can derive the underlying physics from the data.

original date: winter quarter, 2002

1 necessary files

First you need the files. On `PDSF` type

```
cp -r ~romero/template ~/yourdir
```

where `yourdir` is the directory you want to work in.

You now have all the files needed to do the analysis.

2 set up path to production files

Before beginning the analysis, you need the production data. In directory `runs`, we will create a file that points to the data. First locate the data on `PDSF` and do

```
ls /auto/path/to/data/.../somedir/ > runs/runs_something.txt
```

You may have to add a line (indicating the path to the data) to the **.txt** file, and, depending on how the files are stored, you should have a file that looks like this...

```
/auto/stardata/starspec3/flow_pDST_production/reco/minBias22GeVZDC/
2329088
2329089
2329091
2329092
...
```

Where each line represents a directory that contain data files. (Alternatively, you can also list individual files. Put file names in separate lines.)

3 creating histograms

Here's how to obtain dE/dx histograms (50 per run, one for each $m_t - m_0$ slice):
1. edit **y0templatehist.lsf**

- change e-mail in the header (only need to do once), the system will notify you when a batch job is done
- set up the file to use as the data (the **.txt** file you created in previous section)
- set up cuts (very important)

2. now you are ready to run, do
`source yallhistn.scr`
and follow instructions.

In particular, specify the run method: *bsub* is for submitting a batch run (good when you run a lot of files, will run in background); *source* is for running right away (good for a test run with just a few files, you see progress on screen)

commands for batch jobs:
- type `bjobs` to see run status
- type `bkill jobID` to end a job
-[warning: once the job is submitted, and status is still `PEND`, a temporary file HadChgyRapKwd.lsf will be created in the default directory. DO NOT delete this file. The file gets deleted as the job status changes to `RUN`.]

3. once finished, subdirectories and files will be created in `fits` directory, with the name according what you specify in the previous step.

As an example, for the case of 0.1 ¡ y ¡ 0.3, with `ka` mass assumption, and keyword "`test`" - the files will be created in ...

`fits/py.2test/ka` if you are running charge = positive ("`p`")
`fits/ny.2test/ka` if you are running charge = negative ("`n`")

- your histogram file is called → **hist.rootsave**
- typical CPU time ≈ 0.1 sec/event

4 doing fits

Here's how to do Gaussian fits (4 Gaussians per $m_t - m_0$ slice):
1. edit lsf file (created earlier by running hist. For the earlier example, it will be **fits/py.2test/ka/fits.lsf**) for fitting strategies and parameters, and resolution parameter.

2. when you are ready to run, do, for the above example,

```
source fits/py.2test/ka/fits.lsf
```

3. when finished, files will be created in the same directory as the histograms. The file that containts the fits will be called **fits.root**.

4. a **dedx.cal** file is where the calibration parameters go to. The file should include 7 lines, with m1 on line 1, m2 on line 2,..., m7 on line7. The script reads this file to find the centroids of dE/dx Gaussians. You can edit it by hand, but not necessary.

5. to redo the fit, adjust parameters by editing the **.lsf** file and `source` it again.

5 C scripts used in root4star

combinexyz.C is our main script to look up plots that we have created. here are examples:

- to plot histograms,
  `.x combinexyz.C("fits/py0t1/pr/","hist.rootsave","","")`
- to plot fits,
  `.x combinexyz.C("fits/py0t1/pr/","fits.root","","g")`
- to plot yields,
  `.x combinexyz.C("fits/py0t1/pr/","fits.root","","y")`
- to plot ratios,
  `.x combinexyz.C("fits/ny0t1/pr/","fits.root","fits/py0t1/pr/","d","prYield")`

When you run one of the above scripts, the code will spit out the default command. It contains a full set of parameters. You can use this to change slices, adjust x and y ranges and

$$\text{the last number} = \begin{cases} 1 & \text{log scale;} \\ 0 & \text{linear scale;} \\ -1 & \text{show fit parameters (for } \texttt{g} \text{ mode: a = amplitude, c = centroid, w = width)} \end{cases}$$