ROOT Exercises

This is a guide for the ROOT exercise sessions. The exercises cover the following areas:

Session A covers histograms, and the input/output capabilities.

Session B is an example of an analysis using real physics data.

References and Reading Material

To maximize the available time and the access to the ROOT development team we recommend you read parts of the User's Guide before the exercise sessions. To point you to the specific pages we include the pages for each session.

You may also want to read the exercise guide in its entirety to prepare for the exercises.

Subject	Pages in User's Guide
Session A:	
Histograms	75 – 92
Input/Output	98 – 108, 179 – 181, 211
Tree Viewer	181
Event Lists	194 – 208
Session B:	
An Example Analysis	299-303

Session A

In this session you will:

- Fit and rotate a histogram
- Use the Object Browser and the Tree Viewer
- Make a profile and contour graphs
- Build an event list from a cut
- Fill a histograms with random numbers
- Use ACLiC

Fitting a Histogram

Execute the tutorials `hsimple.C` and `hsum.C`.

Question A1: *In the canvas generated by `hsum.C`, how would you add the fit of the second signal `s2` using a Landau distribution in a sub-range? Keep the original histograms are kept on the same picture.*

The Object Browser

Exit the current ROOT session and start a new one. Create a ROOT Browser Object and select the `File:Open` menu item.

```
root[] TBrowser b;
```

Open the file `hsimple.root`. Double click on the ROOT files folder to see `hsimple.root`. Double click on `hsimple.root` and you can see the contents of the file. Find the tree `ntuple` and double click on it. You can see several variables.

Question A2: *How do you histogram the variable `pz` from the `ntuple` in `hsimple.root`.*

Using the Tree Viewer

On the ROOT command line, start the Tree Viewer for the `ntuple`.

```
root[] ntuple->StartViewer();
```

Use the information in the User's Guide on the Tree Viewer (pg. 181) and the `TH1::Draw` options (pg. 81) to answer the next question.

Question A3: *Using the Tree Viewer how would you make a profile histogram of `pz` versus `px`.*

Question A4: *Fit the profile with a polynomial of degree 2.*

Cuts and Contour Graphs

Question A5: *Using the Tree Viewer show `py` versus `px` with the condition that `px^2 + py^2 < 20` using different graphics options (see User's Guide pg. 81).*

Use the option `contz,list` as the final draw option. On the command line reset the global color palette. Notice that the Draw command from the Tree Viewer was printed to the command line. Now you can use the up-arrow to recall the command.

```
root[] gStyle->SetPalette(1)
root[] ntuple->Draw("py:pz","px*px+py*py< 20","contz,list",
25000, 0);
```

Locate the script `FirstContour.C`. Study it to anser the following questions

Question A6: *What is done by these lines?*

```
   TObjArray *contours =
     (TObjArray*)gROOT->GetListOfSpecials()-
>FindObject("contours");
   TList *lcontour1 = (TList*)contours->At(0);
   TGraph *gc1 = (TGraph*)lcontour1->First();
```

Question A7: *What does this loop do in `FirstContour.C`?*

```
while(1) {
      Double_t x = -4 +8*gRandom->Rndm();
      Double_t y = -4 +8*gRandom->Rndm();
      if (cutg->IsInside(x,y)) {
         pm->SetPoint(np,x,y);
         np++;
         if (np == npmax) break;
      }
   }
```

Making an Event List

Use the User's Guide (pg. 198) or the `TTree` class description to look up the syntax to create an event list. From the command line and using the `ntuple` in file `hsimple.root`, generate a `TEventList` with entries having `pz > 10`. Print the lists of events on the screen.

Question A8: *How many events are in the list?*

Use this list to display the `px` for the events in the list.

Look at the history file and see the commands that were generated. From the history file, take the commands that you typed in interactively and make a script.

Question A9: *Write a script to do what you just did interactively: open `hsimple.root`, make an event list with entries having `pz > 10`, print the event list and the number of entries on the screen.*

Question A10: *What is the RMS of the selected `px` distribution?*

Rotating Lego Plots

Execute the tutorial `h1draw.C`. Rotate the Lego histogram in the top right part by left clicking on it and dragging the mouse.

Question A11: *Using a command, how do you get the current viewing angle theta?*

Hint: the viewing angle, theta is an attribute of `TPad`.

Filling Histograms

This next exercise will show you how to fill histograms and take time measurements. Open the file `hrandom.C`. Study how it fills 2 histograms and prints the time it takes to fill them. Execute the script.

```
root [] .x hrandom.C
```

While this is executing, study the script and notice how it is written so it can be compiled.

Question A12: *Copy `hrandom.C` to `hrandom1.C` and modify it to add two more histograms using `TRandom2` and `TRandom3` to fill them. For each case, print the CPU time spent in the random generator.*

Using the Compiler interface ACLiC

Start a new ROOT session and run the same script using ACLiC. You should see a considerable improvement in the CPU time:

```
root [] .x hrandom1.C++
```

Question A13: *Copy the `hrandom1.C` script to `hrandom2.C` and modify it so to draw the histogram and display it after the script has finished.*

Session C

In this session you will:

- Study an example analysis from DESY
- Learn about the `TTree::MakeSelector` method to automatically create a class that loops over each entry in a tree.
- Save and retrieve a canvas to and from a ROOT file
- Compute the integral of a histogram
- Compute the integral of a function within a range

The data are taken from the DESY H1 micro-DSTs. The example data file is in your directory `$H1/dstarmb.root`.

Example Script from `TTree::MakeSelector`

First, you need to understand how the `TTree::MakeSelector` method creates a class that loops over the entries in the `TTree`. Please read Chapter 12 in the User's Guide "Example Analysis" pg.218.

Change directory to `$HOME/tutorials`, the directory you copied in the last session. Now open the script `$HOME/tutorials/h1analysis.C` and read the comments.

The script shows how to use a class (the skeleton has been automatically generated by the `TTree::MakeSelector` method) to loop on the files, perform some cuts and fill two histograms with some physics quantities.

Start a root session and open the file `$H1/dstarmb.root`.

```
root[] TFile f("$H1/dstarmb.root");
```

Using the `TBrowser`, display some of the variables in the `TTree` named `h42`.

```
root[] new TBrowser();
```

Execute the script `h1analysis.C` on the `h42` tree.

```
root[] h42->Process("h1analysis.C")
```

Saving and Retrieving a Canvas

Save the canvas with the dm_d histogram as a ROOT file ("c1.root") with the `File:Save as canvas.root` menu option. Start a new session. Read the saved ROOT file.

```
root[] .q
root
root[] TFile f("c1.root")
root[] c1->Draw()
```

Computing Integrals

Get a pointer to the `Dstar` histogram named "`hdmd`".

Get a pointer to the fitted function "`f5`".

Compute `hint` = the histogram bin integral.

Compute `fint` = the `f5` integral where the function is positive (0.139 – 0.17).

Use the TH1 and TF1 class description web page to find the right method to calculate the integral for each object.

http://cscsrv01/root/root/html/TH1.html

http://cscsrv01/root/root/html/TF1.html

Question C1: *What is the value of* `ratio = fint/hint`*?*