Histograms

Anna Simon

October 28, 2015

No class on Friday, October 30. Instead, please review slides on TGraph and TMultiGraph and complete the posted assignment.

Deleting histograms from the memory

When a pointer is re-declared and set to a new memory location, the access to the location it was pointing to is lost. Memory is allocated but cannot be used. A message "... potential memory leak ..." is printed to the terminal. To print the list of all the objects:

```
gDirectory->GetList()->Print()
```

To delete all the objects:

```
gDirectory->GetList()->Delete()
```

To check if the object exists:

```
gDirectory->GetList()->FindObject("h1")
```

To delete an object:

```
gDirectory->GetList()->FindObject("h1")->Delete()
```

Histogram errors

```
1 h->GetBinError(nBin)
```

will return the error calculated as a sqrt(nEntries) in the given bin
To change the error to the sum of weights squared ($\sum w^2$) set the sumw2 option BEFORE filling in the histogram:

```
1 h->Sumw2() //or h->Sumw2(true), true is the default value
2 h->Sumw2(false) //clears the sumw2 option, error calculated as sqrt(
      nEntries) again
```

- Create an empty histogram h with 10 bins in the range 0, 10
- Fill in the bins from 0-4 with the bin number
- Change the way the errors are calculated by setting sumw2
- Fill in the bins 5-9 with the bin number
- Plot the histogram with error bars (use "e" option).
- Print out the error value for each of the bins.

Scale() and Rebin()

Histogram can be scaled by a value, i.e. the content of each bin will be multiplied by a value:

```
h->Scale(2.3)
```

The histogram can be re-binned, i.e. the size of the bins is increased by a factor and their content is summed into one bin

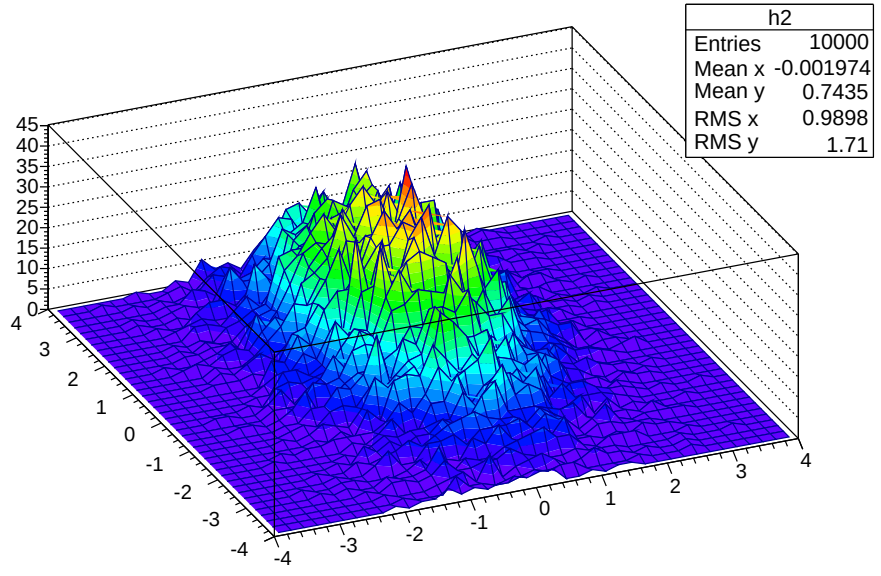
```
TH1F *h = new TH1F("h","",3000,0,3000)
h->Rebin(3)
//now h has 1000 bins in the region 0 to 3000
```

NOTE: the binning factor has to be an exact divider of the original number of bins (for this purpose having the number of bins in the powers of 2 is useful) NOTE2: rebinning is irreversible. To avoid overwriting the original histogram, save the rebinned one into another histogram:

```
TH1 * hReb = h->Rebin(4,"hReb");
```

2-D histograms

```
1 TH2D *h2 = new TH2D("h2", "Example 2D histo", 40,-4,4,40,-4,4);
2
3 for (int i = 0; i<10000; i++) {
4    double x = gRandom->Gaus(0,1);
5    double y = gRandom->Gaus(1,2);
6    h2->Fill(x,y);
7 }
8
9 h2->Draw(); //options colz, contour, lego(1-3), text, surf(1-5)
```



Projections and profiles

A 2D histogram can be projected to one of the axis:

```
1 h2->ProjectionX("hProjX",firstBin,lastBin)
```

The bins with the same x value are summed together.

A 2D histogram can be projected to one of the axis:

```
1 h2->ProfileX("hProfileX",firstBin,lastBin)
```

For all the bins with the same x value a mean of all the y values is calculated.

Problem 1: 1-D histograms

- Make a gaussian filled histogram h1 between 0 and 10 with 100 bins and 1000 entries with mean 5 and sigma 1.
- Make another histogram h2 uniformly distributed between 0 and 10 with 100 bins and 10000 entries.
- Add the two histogram into a new one using TH1::Add (hSignal = h1 + h2)
- Make another histogram hBgrd, containing a uniform distribution still with 100 bins but with 100000 entries. Normalize this histogram to have a total integral of 10000 using TH1::Scale.
- Subtract hBgrd from hSignal using TH1::Add (hDiff = hSum - hBgrd)
- Plot the result using the error option (hDiff->Draw("E")). Do the error make sense? If not, how can you get the correct bin errors

Problem 2: 2-D histograms

- Create a 2 dimensional histogram with x and y in the range [-5,5] and [-5,5] and 40 bins in each axis.
- Fill the histogram with correlated random normal numbers. To do this generate 2 random normal numbers (mean=0, sigma=1) u and w. Then use $x = u$ and $y = w + 0.5 * u$ for filling the histogram.
- Plot the histogram using plot option you prefer.
- After having filled the histogram, compute the correlation using TH1::GetCorrelationFactor.
- Make a projection of the 2-dimensional histogram on the y-axis.