ULHISTOS

histogramming package for Ulysses

Version 3.1

Luis Peralta

Lisbon, October 2012

Table of Contents

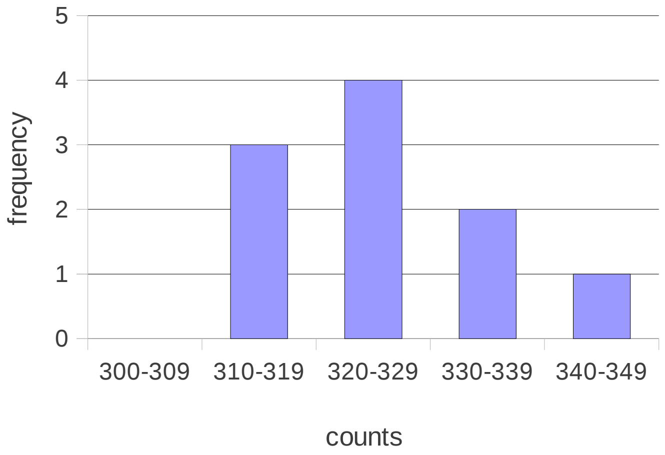
Introduction

The package allows for simple histogramming and is inspired in the HBOOK package developed at CERN [1]. Apart from the basic booking, filling and output of histograms, the package also allows a few basic operations on histograms and gives the user the ability to retrieve information from the histograms. The histogramming package is part of the ULYSSES geometry and tracking package for particle transport.

The original program was written in FORTRAN 77. Presently, a FORTRAN 95 compiler must be used. The ULHISTOS package is open-source software, released under the GNU *General Public License*.

Histogramming basics

Histograms are a kind of plot were the data enters in a discrete way. The x axis is divided in a certain number of channels or "bins" . Each one in will then contain a quantity, which in the simplest case will be the number of times the value x occurred in that interval. In this case the histogram represents the frequency distribution of variable x. As an example suppose one has a Geiger counter and wants to get the counts distribution during a fix acquisition time. Each time a count x is obtained with the Geiger an event is added in the channel corresponding to interval where x lies. Suppose 10 acquisitions are made with the values 311, 325, 320, 331, 317, 322, 330, 315, 340, 328, and the histogram has channels spanning over the following intervals 300-309, 310-319, 320-329, 330-339, 340-349. The histogram frequencies (or channel contents) will be c1=0, c2=3, c3=4, c4=2, c5=1 and the histogram would look like the one in the figure below:



Histogram with 10 entries.

Some of the histogram basic characteristics are:

The histogram bounds:

Lower and upper bounds (x_{min} and x_{max}) must be chosen for each histograms. They correspond to the smallest and biggest expected values. Should a x value occur with smaller ou bigger value than x_{min} and x_{max} that value will not be recorded in the histogram. Underflow and overflow counters are provided for each histogram to keep the account of such events. Inspecting this counters the user takes knowledge of these events and can take the appropriate action: normally choosing a new x_{min} or x_{max} value and run the program again.
Remember, the overflow and underflow counters tells how many events are above or below histogram bounds and not the accumulated quantities.

The number n of channels in which the variable x will be divided:

This number will limit the histogram resolution. Dividing the full x range in n intervals results in loosing information on the actual value x value. The higher number of n channels, the best x resolution and more accurate is the information on the actual x. The price to pay is a bigger dispersion of the data over the channels, resulting on a bigger statistical uncertainties in each channel. So the choice of n is compromise between resolution and statistical uncertainty.

The bin size:

The channel width or bin size will be set equal for all channels and is defined as

$$\Delta x = \frac{x_{max} - x_{min}}{n}$$

The channels contents:

In each channel the entered quantities q are summed. If each entry is equal to one, then the channel content $c_i = \sum q$ is simply its occurrence frequency. In this case the total number of entries equals the histogram integral. In general, the scored quantities are different from the unity and the histogram integral is obtained summing up all channel contents, $S = \sum_{i=1}^{N} c_i$. The distribution integral would be, the product of this integral by the bin size (for equal bin sizes), i.e.

$$I = \int_{x_{min}}^{x_{max}} f(x) dx \approx S \Delta x = \sum_{i=1}^{N} c_i \Delta x \quad .$$

Histogram types

The package main purpose is to score the results from particle tracking simulations, in particular the ones using the Ulysses package. Three histogram types have been foreseen for the scoring of the most common variables.

Type 1: Frequency histograms:

This type of histogram is provided for scoring the number of counts of a particular variable x. In most cases the filling routine will be called to score one count (q=1), but any number of counts q can be scored at each call. An example is the energy spectrum obtained from a detector. For each event *j* the total deposited energy *E_j* is computed and at channel *i* corresponding to this energy a quantity equal to 1 is added.

Let's analyse this example in more detail:

At booking time the user decides what is the energy range *[E_{min},E_{max}]* for the histogram and the number *n* of channels in which this interval will be divided. Each channel will have a size of

$$binz=\frac{E_{max}-E_{min}}{n}$$

For each event *j* a total energy E_j is deposited and is scored (added) in channel *i* given by

$$i=\mathrm{int}\left[\frac{(E_j-E_{min})}{binz}\right]+1$$

where the Fortran function *int(x)* returns the greatest *integer* less than or equal to *x*.

A quantity *q=1.0* (one event) is then added to the existing content of channel *i*.
The final content (or score) *f(i)* of channel *i* over ***Nev*** events is given by

$$f(i)=\sum_{j=1}^{Nev} q_j$$

which will be simply the number of entries in that channel if $q_j=1$ for all events.
If the quantity q is a counting then the uncertainty for each channel will be $\sigma_f^2(i)=\sum\sigma^2(q_j)=\sum q_j$ and thus $\sigma_f(i)=\sqrt{f(i)}$ is the usual value for the uncertainty of a *number of counts f.*

If events with weights w_j are being used then the content in the channel *i* will be given by

$$f(i)=\sum_{j=1}^{Nev} q_j w_j$$

and since $\sigma_f^2(i)=\sum\sigma^2(q_j w_j)=\sum\sigma^2(q_j)w_j^2$ because w_j are constants without uncertainty we have

$$\sigma_f(i)=\sqrt{\sum q_j w_j^2}\ .$$

Type 2: Event Average Histograms

The second type of histograms arises when the content of each channel is not a number of counts but the average value of some physical variable for a given number of events *Nev*. The most common example is the energy deposited by a particle beam as a function of some spatial variable (e.g. depth *z*). For each event ***j*** the primary and secondary particles will deposit small amounts of energy δE in the location between z and $z+\Delta z$ corresponding to the *i* channel of the histogram. Thus a particular channel ***i*** might be filled several times with different δE amounts during the same event simulation and these amounts will add up to $\Delta E(i)=\sum \delta E=Q(i)$. For a beam of N_{ev} primary particles (or events) the average deposited energy $E(i)$ in channel *i* ***per event*** will by given by

$$\bar{E}(i)=\frac{\sum_{j=1}^{Nev} \Delta E_j(i)}{N_{ev}}$$ or for a general variable one defines

$$\overline{Q(i)}=\frac{\sum_{j=1}^{Nev} Q_j(i)}{N_{ev}} \quad \text{and} \quad \overline{Q^2(i)}=\frac{\sum_{j=1}^{Nev} Q_j^2(i)}{N_{ev}}$$

and its uncertainty is given by

$$\sigma(\overline{Q}(i))=\frac{\sqrt{\overline{Q^2(i)}-(\overline{Q(i)})^2}}{\sqrt{N_{ev}}}$$

If each event has a weight w_j one has

$$\overline{Q(i)}=\frac{\sum_{j=1}^{Nev} Q_j(i) w_j}{\sum_{j=1}^{Nev} w_j} \quad \text{and} \quad \overline{Q^2(i)}=\frac{\sum_{j=1}^{Nev} Q_j^2(i) w_j}{\sum_{j=1}^{Nev} w_j} \quad .$$

The uncertainty on the average is still given by the former equation.

Type 3: Channel Average Histograms

In some cases we are interested in the average value of an input variable relative to the events entered in that particular channel (bin).

As an example suppose one has an array of detectors and is interested in the average energy deposited in each one of the individual detectors by the particles hitting that particular detector, and not all the particles hitting the array. In this case if *$N_{bin}(i)$* is the number of particles hitting the channel (detector) ***i*** and $\Delta E_j(i)$ is the deposited energy in that detector, the average deposited energy over there will be

$$\bar{E}(i)=\frac{\sum_{j=1}^{N_{bin}(i)} \Delta E_j(i)}{N_{bin}(i)} \quad .$$

For a general variable Q we will have

$$\overline{Q(i)}=\frac{\sum_{j=1}^{N_{bin}(i)} Q_j(i)}{N_{bin}(i)} \quad \text{and} \quad \overline{Q^2(i)}=\frac{\sum_{j=1}^{N_{bin}(i)} Q_j^2(i)}{N_{bin}(i)}$$

and the uncertainty will be

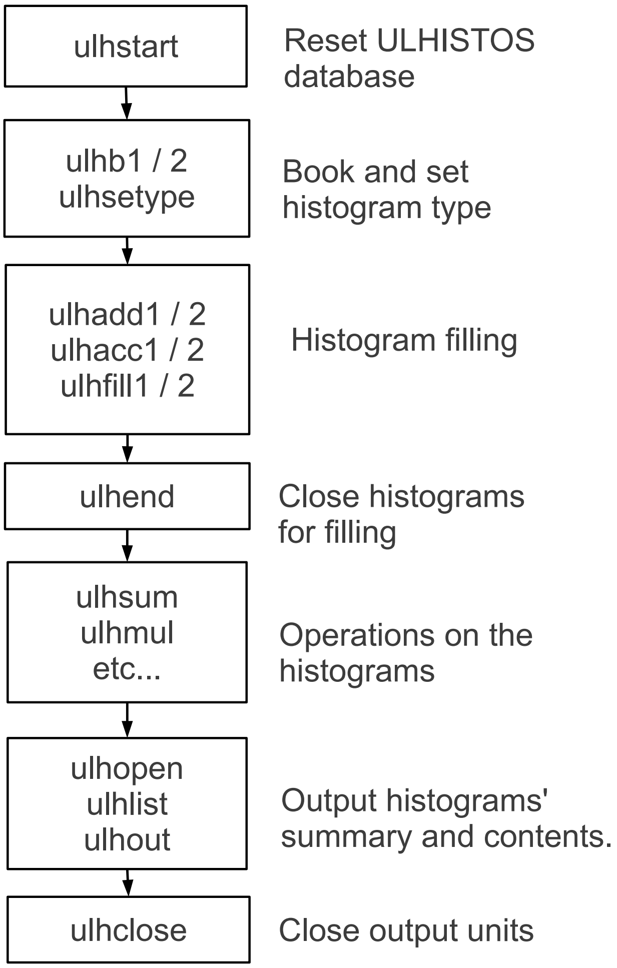
$$\sigma(\overline{Q}(i))=\frac{\sqrt{\overline{Q^2(i)}-(\overline{Q(i)})^2}}{\sqrt{N_{bin}(i)}}$$

In the case of weighed events we have

$$\overline{Q(i)}=\frac{\sum_{j=1}^{N_{bin}(i)} Q_j(i) w_j}{\sum_{j=1}^{N_{bin}(i)} w_j} \quad \text{and} \quad \overline{Q^2(i)}=\frac{\sum_{j=1}^{N_{ev}(i)} Q_j^2(i) w_j}{\sum_{j=1}^{N_{ev}(i)} w_j}$$

Package usage

A typical usage could be the following: to start, a call to routine **ulhstart** is made to reset the ULHISTOS database. The histograms will be booked using the appropriate routines and according to their dimension. The routine **ulhsetype** is used to set the histogram type (frequency, event average or channel average histogram). The histograms will be filled using one of the filling routines and closed in the end of the simulation. After the end of filling the histograms several operation can be made on the histograms, like summing up two histograms or obtain information on statistical parameters (average and standard deviation, etc.). Finally the histograms will be outputted.



The package structure

The Database

The 1-d and 2-d histograms are stored in arrays in two separate commons, **ulhistos1** and **ulhistos2.** The number of allowed histograms is controlled by the parameters **nh1d** and **nh2d** set by default to 100 and 10. The histograms ID number can however be chosen in the range 1 to **npoint** (the number of pointers) which is set by default to 9999. The maximum number of channels in each histogram is controlled by the parameters **nmax1d** and **nmax2d** which are set to 10000 and 200 respectively for 1-d and 2-d histograms. This values can be changed in the include file "**ulhistos.inc**". The size of the **ulhistos1** and **ulhistos2** commons is approximately equal to **nh1d* nmax1d*2** and **nh2d*nh2d*nmax2d*2**.
At filling time the arrays **histo1d** and **histo2d** will contain the histogram contents, while the arrays **histo1dv** and **histo2dv** will contain the contents squared. After calling the routine **ulhend** this array will contain the variable statistical uncertainty.

For most applications, the standard user will have no need to directly change the variables in the **ulhistos.inc** file. All the communication is done through the variables passed in the routines.

Data output

Each histogram is output to a separate file. This allow the direct use of GNUPLOT [2] to display the data. The output is also easily included in a spreadsheet.

The reserved output/input units are kept in the variables of common/units and the defaults are:
iue= 6 : output of program error messages to the standard terminal
ius= 6 : output of histograms summary list to the standard terminal
iuo=99 : output of histogram contents
iui=81 : input unit

In Linux the output to a logical unit=x will result in the creation of a file named **fort.x** where the information is written. To change the file name where output is directed the routine **ulhopen** is provided (see an example programs).

Routine usage

Set and Reset routines

The first routine to be called by the user must be **ulhstart**. This routine resets the database values, and initializes the standard units for output.

Individual histograms can be reset using **ulhreset**. This routine will reset the histogram contents, number of entries, underflow and overflow, but will not change the histogram type or dimension. In fact, no routine is provided to delete histograms. So, once defined an histogram *id*, that value can not be used for other histogram *id* with other characteristics.

A different histogram title can be set with **ulhsetitle**, which is particularly useful when the histogram is a copy of another.

The histogram type is set using **ulhsetype** (see below).

Histogram booking

To create an histogram, its characteristics must be passed to the program in the first place. The lower and upper bounds $\mathbf{x_{min}}$ and $\mathbf{x_{max}}$ the number of channels **n** and a title will be required. The histogram is identified by an ***id*** number which is unique for each histogram and is set by the user. The routines **ulhb1** and **ulhb2** (for 1 and 2 independent variable histograms) are provided for this purpose and must be called prior to the use of the histogram.

The histogram type is declared with **ulhsetype** after histogram booking. Three different types of histograms are foreseen:

1- Frequency Histogram: The content of each channel is the sum of the input quantities. This is the default histogram type.

2- Event Average Histogram: During the event **iev** the quantities falling into a particular channel i are summed. After the histogram is closed for filling the average by the number of all events **Nev** is obtained. This type is specially devised to score the average energy deposited as a function of another variable (e.g. depth or radial distance to some axis).

3- Channel average histogram: during filling the content of each channel is the sum of the input quantities. After the histogram is closed for filling in each channel the average per number of channel entries is obtained.

An histograms can be a copy of an existing histogram, and the routine **ulhcopy** is provided for that purpose. The routine is specially useful when a series of histograms all with the same characteristics must be defined.

An histogram can be booked and filled with an external (one dimensional) function using the routine **ulhbf1**.

Histogram filling

In Monte Carlo programs, histograms are filled on an event by event basis. At a given event, for a variable *x* a quantity *q* will be summed in channel *i* to the already existing content of that channel. The operation will be repeated over and over, many times until the end of the program.

Frequency histograms (type 1):

For frequency histograms and unweighed events, the routines **ulhadd1** and **ulhadd2** are provided for filling.

If events with weights w_j are being used the routines **ulhfill1** or **ulhfill2** must be used. These two routines are the general routines for histogram filling but they are slower than the routines provided for unweighed events. Their use should be avoid if possible.

Event Average histograms (type 2):

For type 2 histograms the routines **ulhacc1** and **ulhacc2** are provided for filling for unweighed events. To use events with weights w_j the routines **ulhfill1** or **ulhfill2** must be used as explained.

Channel average histogram (type 3):

These histogram type must be filled using ulhave1 or ulhave2 routines, for unweighed events. Again routines **ulhfill1** or **ulhfill2** are used for weighted events.

Closing the histogram

After filling histograms must be closed by calling **ulhend**. This routine will ensure the uncertainties are correctly computed.

Operations on histograms

After filling the histograms and before output some arithmetic operations can be performed on the histograms. The operation are performed on histograms with the same characteristics (dimension, number of channels and lower and upper bounds). The operation is made on a channel-by-channel basis. The result is always a new histogram (with different id). The available operations are:

ulhsum: add two histograms
ulhmul: multiply two histograms
ulhdiv: ratio of two histograms
ulhscale: multiply the histogram by a scaling factor
ulhsqrt: get the contents square root
ulhexp: get the exponential of the contents
ulhlog: get the logarithm of the contents

Input/ Output the results

Routine **ulhout** is provided to output the histogram data to an ASCII file. Histograms can be piled up in a single file or distributed by several files. To open a file with a name chosen by the user for the output of the histograms the routine **ulhopen** must be called prior to **ulhout**.

Getting an histogram summary

The routine **ulhlist** provide a summary of the histograms and may be called at any time during the program. To output the information to a user file a call to **ulhopen** must be made before the call to **ulhlist**. For 1-d histograms the following information is written in the declared output unit:

histogram title
ID
type
status
number of channels
lower bound xmin
upper bound xmax
bin size
total number of entries
number valid entries
sum of contents
underflow
overflow

For 2-d histograms information on the y number of channels, ymin, ymax and y bin size is also given.

For histogram input, three routines are provided. Routines **ulhin1** and **ulhin2** input data from a file with the same format as the output used by **ulhistos** (ie with 3 columns). The purpose of these routines is to allow the input of data already processed by another program. Another input routine, **ulhinsp** is provided for the input of spectra and other data organized in two columns (channel, content). The first line of this file is assumed to be the title. Data must start at the second line.

Retrieving information

The routine **ulhinfo** retrieves information from the histogram and can be used any time during the program execution. The information is filled into an array and is the following:

1- nx, number of channels in x
2- xmin , lower bound in x
3- xmax , upper bound in x
4- binx , channels size in x
5- ny , number of channels in y
6- ymin , lower bound in y
7- ymax , upper bound in y
8- biny , channels size in y
9- nentries , total number of entries
10- underflow , number of underflow events
11- overflow , number of overflow events
12- sum of contents
13- dimension (1 or 2)
14- status (1,2,3,4 or 5)
15- type (1,2 or 3)
16- number of events within bounds

For frequency histograms the distribution *mean value, standard deviation and mean uncertainty* can be obtained from **ulhxmean**. If instead of the histogram full range only, those values are to be computed on a more restricted interval then the routine **ulhxm** should be used.

If simple peak analysis is required the routine **ulhpeak** for a range [x_{min}, x_{max}] will provide the peak centroid, peak width (standard deviation), total peak area and net area after background subtraction. A linear background is assumed in the peak range.

A set of routines are provided to access the information at a particular channel. The routines **ulhgetcx** and **ulhgetcxy** will get the content and error for given *x* value or point *(x,y)*. If the channel number *i* or *(i,j)* available the routines **ulhgetci** or **ulhgetcij** can be used with the same purpose.

The channel i ((i,j)) corresponding to x ((x,y)) can be retrived using **ulhgeti** (**ulhgetij**). The routines **ulhgetx** and **ulhgetxy** are provided to retrieve the x or (x,y) value at the channel center given the channel number i or (i,j).

The routines **ulhgetmax1** and **ulhgetmin1** will compute maximum and minimum content value respectively for the histogram.

Two routine are provided to directly input a value into an histogram channel. These routines are ulhputci and **ulhputcij** for 1 and 2 dimension histograms, respectively.

Routine description

Booking and reset histograms

****

ulhstart

****

Purpose: reset database

This routine must be the first one to be called. All database variables are reset to zero and standard units are set to the default values: iue=6, ius=6, iuo=99, iui=81.

****

ulhb1(id,title,nx,xmin,xmax)

ulhb2(id,title,nx,xmin,xmax,ny,ymin,ymax)

****

Purpose: Booking of 1-d and 2-d histograms

id- histogram id number : *integer*4*

title- histogram title : *character*64*

nx- number of channels for x : *integer*4*

xmin- lower bound for x : *real*8*

xmax- upper bound for x : *real*8*

ny- number of channels for y : *integer*4*

ymin- lower bound for y : *real*8*

ymax- upper bound for y : *real*8*

Histogram properties are set in this routines. The histogram **id** is a number between 1 and 9999. This default can be change by modifying parameter **npoint** in the **ulhistos.inc** file. The **id** number is unique to each histogram. The title can have up to 64 characters, and is used as an information to the user. The maximum allowed number of channels is 10000 for 1-d histograms and 200 for 2-d histograms. The default value can be changed in parameters **nmax1d** and **nmax2d** in the **ulhistos.inc** file. On calling the routine, all channels contents are reset as well as the number of entries, overflow and underflow counters. Histogram status flag is set to zero. This is default status for filling the histograms. The *histogram status flag* is set to 1 on calling routine **ulhend** for *type 1* histograms and is set to 2 for *type 2* and *3* histograms. If the user sets the contents and errors with **ulhputci** or **ulhputcij** the status flag is set to 3. Also when calling **ulhin1** the status flag is set to 3. The status of an histogram created by **ulhbf1** is set to 4. After calling **ulhrndi** the status is set to 5.

****

ulhcopy(id1,id2)

****

Purpose: Copy of histograms

id1: id number of original histogram : *integer*4*

id2: id number of the copy : *integer*4*

Histogram id1 contents and errors are copied to histogram 2. Other variables like the title, number of channels, lower and upper bounds, overflow, underflow, number of entries and status flag are also copied. Histogram id2 must not exist, or an error message will be issue and the copy will no take place.

****

ulhbf1(id,title,nx,xmin,xmax,fun)

****

Purpose: Booking and fill a 1-d histogram with an external function

id- histogram id number : *integer*4*

title- histogram title : *character*64*

nx- number of channels for x : *integer*4*

xmin- lower bound for x : *real*8*

xmax- upper bound for x : *real*8*

fun- 1-d function must be declared EXTERNAL in the calling routine

Book and fill an histogram with external function fun(x). The routine is a mixture of booking and filling routine.

****
ulhreset(id)
****
purpose: Reset contents of an histogram
id- histogram id number: *integer*4*

The contents, errors, number of entries, overflow, underflow and status flag are reset. The title is not reset.

****
ulhsetype(id,itype)
****
purpose: Set the histogram type
id- histogram id number : *integer*4*
itype- histogram type : *integer*4*

Histogram types

itype=1 Frequency histogram (default)
itype=2 Event average histogram
itype=3 Channel average histogram

By default all histograms are set to itype=1 at **ulhstart**

If **id=0**, all booked histograms are set to type=itype

****
ulhsetitle(id,title)
****
purpose: Set the histogram title
id- histogram id number: *integer*4*
title- histogram title : *character*64*

Allow the histogram title to be changed. Useful when the histogram is a copy or has been reset.Filling the histograms

****
ulhadd1(id,x,q)
ulhadd2(id,x,y,q)
****

purpose: Histogram filling type 1 histograms

id- histogram id number : _integer*4_

x- position x : _real*8_

y- _position y : real*8_

q- additive quantity : _real*8_

A quantity q is added at position x for histogram id. A quantity q2 is added to the error array at position x.

****
ulhacc1(id,x,q,iev)
ulhacc2(id,x,y,q,iev)
****

purpose: Filling of type 2 histograms (event average)

id- histogram id number : _integer*4_

x- position x : _real*8_

y- _position y_ : _real*8_

q- additive quantity : _real*8_

iev- present event number : _integer*4_

This routines are intended to be use for **type 2** event average histograms, where for a given event **iev,** each position x might be filled more than once. The routine uses partial counters where the quantities q are accumulated during event **iev**. Upon a new event, the contents of the partial counters are summed to the arrays **histo1d(2d)** for the contents and **histo1dv(2dv)** for the contents squared. To get the average and uncertainty after filling, routine **ulhend** must be called. The quantity q can be positive or negative

****

ulhave1(id,x,q)
ulhave2(id,x,y,q)

****

purpose: Filling of type 3 histograms (channel average)

id- histogram id number : *integer*4*

x- position x : *real*8*

y- *position y* : *real*8*

q- additive quantity : *real*8*

This routines are intended to be use for **type 3** channel average histograms. After the call to ulhend the histogram contents $\sum q$ are substitute by the channel average $\sum q/N_{bin}$ where N_{bin} is the number of entries in that channel.

****

ulhfill1(id,x,q,wght,iev)
ulhfill2(id,x,y,q,wght,iev)

****

purpose: General filling routine

id- histogram id number : *integer*4*

x- position x : *real*8*

y- *position y* : *real*8*

q- additive quantity : *real*8*

wght- event weight *:real*8*

iev- present event number : *integer*4*

These are the general filling routines, which can be used with any type of histogram. Type 3 histograms can only be filled using these routines. The weight **wght** is global to a particular event **iev** (i.e. must not change during the event). *This is a slow routine. If no weights are used the other filling routines should be used.*

Input / Output routines

****
ulhin1(id)
****

Purpose: input one 1-d histogram from a file

d- histogram id number : *integer*4*

The histogram **id** is read from the input unit (by default iui=81). The file name and unit number can be changed using the routine **ulhopen**.

The format of the file must be the one used by **ulhout** (see below).

If histogram **id** exists the routine resets it. Otherwise this routine will book histogram **id**.

The status flag is set to 3.

****
ulhin2(id,nx,nx)
****

Purpose: input 2-d histogram from a file

d- histogram id number : *integer*4*

nx- number of x channels : *integer*4*

ny- number of y channels : *integer*4*

The histogram **id** is read from the input unit (by default iui=81). The file name and unit number can be changed using the routine **ulhopen**.

The format of the file must be the one used by **ulhout** (see below).

If histogram **id** exists the routine resets it. Otherwise this routine will book histogram **id**.

The status flag is set to 3.

****
ulhinsp(id)
****

Purpose: input a spectrum from a file

d- histogram id number : *integer*4*

The spectrum is read from the input unit (by default iui=81), which must be already open. The file name and unit number can be changed using the routine **ulhopen**.

The format of the file must the following:

1st line - title (up to 64 characters)

other lines with two columns, real numbers

****

ulhout(id)

****

Purpose: Output one histogram to a file

d- histogram id number : *integer*4*

The histogram id is written to the output unit (by default iuo=99). The file name and unit number can be changed using the routine **ulhopen**. The output is formatted in such a way that can be used by Gnuplot [2]. On the top of the file the histogram id and title is written. The line begins with a # to comply with the Gnuplot rules (comment lines begin with a # in Gnuplot macros). A second line is written with

```
“#     x         content     error "   or
“#     x           y         content     error"
```

(depending if the histogram is 1-d or 2-d. Each column has a width of 12 characters and is separated by a blank. The numbers are written with the format 3(1x,e12.6) or 4(1x,e12.6). In the case of 2-d histograms, for each x value all y values are output, and a blank line is left to mark the next x value.

****

ulhlist(id)

****

Purpose: list histogram summary

id- histogram id number : *integer*4*

Th summary list of the histogram is written to the summary unit (ius=6 the standard output unit - the terminal, by default). If **id=0** then the summaries of all histograms are written to that output unit.

****

ulhoutls(id,ip,itype)

****

Purpose: Output histogram summary list

id- histogram id number : *integer*4*

ip- histogram id number pointer : *integer*4*

itype- histogram type : *integer*4*

This routine is called by **ulhlist**. The routine **ulhoutls** ***is not*** intended to be called by the user.

The output information is the following:

histogram title
histogram ID number
histogram type (1, 2 or 3)
histogram status (0, 1, 2, 3, 4 or 5)
x number of channels
y number of channels (only 2d histograms)
xmin: x lower bound
xmax: x upper bound
x channel size
ymin: y lower bound (only 2d histograms)
ymax: y upper bound (only 2d histograms)
y channel size (only 2d histograms)
number of entries
number of valid entries (entries not under or overflow)
sum of contents
number of underflow entries
number of overflow entries

****
ulhopen(iunit,filename,utype)
****
Purpose: Open a file for output

iunit- unit number : *integer*4*

filename- file name : *character*64*

utype- unit type : *character*1* (**'e'** for error messages, **'s'** for summary lists, **'o'** for output histograms and **'i'** for input histograms)

The file "filename" will be open for error messages/ summary lists / output histograms /input histograms depending on **utype.** The routine allows the user to change the name of the output unit, which otherwise will be the standard output (terminal) or for the histogram output in linux will have the name **fort.99** . The files are opened with fortran status "**unknown**". If the files doesn't exist a new one is created. Otherwise its contents will be superseded by the new ones. It's thus a good idea to change the file names, to avoid losing data.

****
ulhclose
****

Purpose: Close all units

This routine must be called at end of the main program

Operations on histograms

****
ulhsum (id1,id2,id3,a,b)
****

Purpose: Sum of two histograms multiplied by constants

id1- first histogram id : *integer*4*

id2- second histogram id : *integer*4*

id3- result histogram id : *integer*4*

a- first constant : *real*8*

b- second constant : *real*8*

The contents and errors **q** of each histogram **id3** channel are obtained as

$$q_3 = aq_1 + bq_2 \quad \text{and} \quad \delta q_3 = \sqrt{a^2(\delta q_1)^2 + b^2(\delta q_2)^2} \ .$$

Histogram **id3** must be a new histogram. If **id3** exists an error message is issued and the operation is not completed.

****
ulhmul(id1,id2,id3,a)
****

Purpose: Multiplication of two histograms multiplied by a constant

id1- first histogram id : *integer*4*

id2- second histogram id : *integer*4*

id3- result histogram id : *integer*4*

a- constant : *real*8*

The contents and errors **q** of each histogram **id3** channel are obtained as

$$q_3 = a\ q_1 \times q_2 \quad \text{and} \quad \delta q_3 = a\ \sqrt{q_2^2(\delta q_1)^2 + q_1^2(\delta q_2)^2} \ .$$

Histogram **id3** must be a new histogram. If **id3** exists an error message is issued and the operation is not completed.

****

ulhdiv(id1,id2,id3,a)

****

Purpose: Division of two histograms multiplied by a constant

id1- first histogram id : *integer*4*

id2- second histogram id : *integer*4*

id3- result histogram id : *integer*4*

a- constant : *real*8*

The contents and errors **q** of each histogram **id3** channel are obtained as

$$q_3 = a\,\frac{q_1}{q_2} \quad \text{and} \quad \delta q_3 = a\sqrt{\frac{(\delta q_1)^2}{q_2^2} + \frac{q_1^2(\delta q_2)^2}{q_2^4}}\ .$$

Histogram **id3** must be a new histogram. If **id3** exists an error message is issued and the operation is not completed.

****

ulhscale(id1,id3,a)

****

Purpose: Multiplication of an histogram by a constant

id1- first histogram id : *integer*4*

id3- result histogram id : *integer*4*

a- constant : *real*8*

The contents and errors **q** of each histogram **id3** channel are obtained as

$$q_3 = a\, q_1 \quad \text{and} \quad \delta q_3 = a\,\delta q_1\ .$$

Histogram **id3** must be a new histogram. If **id3** exists an error message is issued and the operation is not completed.

****

ulhsqrt(id1,id3)

****

Purpose: Square root of an histogram

id1- first histogram id : *integer*4*

id3- result histogram id : *integer*4*

The contents and errors **q** of each histogram **id3** channel are obtained as

$$q_3=\sqrt{q_1} \quad \text{and} \quad \delta q_3=\frac{1}{2}\frac{\delta q_1}{q_1} \quad .$$

Histogram **id3** must be a new histogram. If **id3** exists an error message is issued and the operation is not completed.

****

ulhexp(id1,id3,b)

****

Purpose: Exponencial of an histogram

id1- first histogram id : *integer*4*

id3- result histogram id : *integer*4*

b- constant *: real*8*

The contents and errors **q** of each histogram **id3** channel are obtained as

$$q_3=\exp(b\,q_1) \quad \text{and} \quad \delta q_3=b\ q_1\ \delta q_1 \quad .$$

Histogram **id3** must be a new histogram. If **id3** exists an error message is issued and the operation is not completed.

****

ulhlog(id1,id3)

****

Purpose: Natural logarithm of an histogram

id1- first histogram id : *integer*4*

id3- result histogram id : *integer*4*

The contents and errors **q** of each histogram **id3** channel are obtained as

$$q_3 = \ln q_1 \quad \text{and} \quad \delta q_3 = \frac{\delta q_1}{q_1} \; .$$

Histogram **id3** must be a new histogram. If **id3** exists an error message is issued and the operation is not completed.

Histograms transformation

****

ulhend(id)

****

Purpose: Ends (closes) the histogram for filling

id- histogram id : *integer*4*

If **id=0** then all histogram filling is terminated.

After **ulhend** is called the uncertainties of each channel are computed and filling is no long possible but other operations can be made on the histograms. This routine calls the internal routine **ulhendid** which in terms selects the appropriate end routine **ulhaddend, ulhaccend, ulhaveend**, according to the histogram type.

****

ulhendid(id)

****

Purpose: selects the appropriate end routine

id- histogram id : *integer*4*

This routine is not intended to be called by the user.

****

ulhaddend(id)

****

Purpose: Close type 1 histograms for filling.

id- histogram id : *integer*4*

User must not call directly this routine but **ulhend** instead.

This routine is used to end type 1 histograms. After calling this routine the histogram status changes to 1. Uncertainties are computed in this routine.

****
ulhaccend(id)
****

Purpose: Close type 2 histograms for filling.

id- histogram id : *integer*4*

User must not call directly this routine but **ulhend** instead.

This routine is used to end type 2 histograms. Uncertainties are computed in this routine. After calling this routine the histogram status changes to 2.

****
ulhaveend(id)
****

Purpose: Close type 3 histograms for filling.

id- histogram id : *integer*4*

User must not call directly this routine but **ulhend** instead.

This routine is used to end type 3 histograms. Uncertainties are computed in this routine. After calling this routine the histogram status changes to 2.

Using the information in the histograms

****

ulhinfo(id,hinfo)

****

Purpose: Get information on histogram id

id- histogram id : *integer*4*

hinfo(16)- array with info : *real*8*

The information on several variables are passed to the array **hinfo**. The array elements are real*8. In the case of variables of integer nature (like **nentries** the number of entries in the histogram) the user should use the function int to convert the value (ex. nentries=int(hinfo(9)). The assignment of each array element is the following:

1- **nx**

2- **xmin**

3- **xmax**

4- **binx**

5- **ny**

6- **ymin**

7- **ymax**

8- **biny**

9- **nentries**

10- **underflow**

11- **overflow**

12- **sum of contents**

13- **dimension (**1= one variable, 2= two variables)

14- **status** (0=not defined, 1=frequency histo, 2=average histo, 3=errors set by user,
4=function, 5=rnd generator)

15-**type** (1- Frequency, 2-Average, 3-Channel average)

16-**nevent** (number of events. This is different from nentries for type 2 and 3 histograms)

****

ulhxmean(id,xmed,sigma,sx)

****

Purpose: Get content mean value, sigma and error on mean of a type 1 histogram

id- histogram id : *integer*4*

xmed- x mean value : *real*8*

sigma- distribution standard deviation : *real*8*

sx- error on mean value : *real*8*

This routine computes the mean value of the x variable, its standard deviation an statistical error on the x variable. The contents c_i of the histogram are used as the (non-normalized) probability distribution. The histogram is thus assumed to be of the frequency type, so c_i represents the number of entries (frequency) of channel i . The sum of all c_i is equal to the number of valid entries in the histogram (i.e. between the lower and upper x values). For an histogram with underflow or overflow this number (the sum of contents) will not be equal to the number of entries!

The routine should be used only for histogram satisfying the following conditions:

All event have weight **wght=1**

All score quantities are equal to **1** (ie q=1)

Histogram is type **1**

The variables are computed according to:

$$\mathbf{xmed} = \frac{\sum (x_i c_i)}{\text{total}} \quad \text{where} \quad \text{total} = \sum c_i$$ is the sum of contents over all channels,

$$\mathbf{sigma} = \sqrt{\frac{\sum x_i^2 c_i}{\text{total}} - \text{xmed}^2} \quad ,$$

and the uncertainty sx on the mean value xmed is

$$\boldsymbol{sx} = \frac{\boldsymbol{sigma}}{\sqrt{total}} \quad .$$

****

ulhxm(id,xmin,xmax,xmed,sigma,sx)

****

Purpose: Get mean value, sigma and error on mean, in a range

id- histogram id : *integer*4*

xmin -x lower value : *real*8*

xmax- x upper value : *real*8*

xmed- x mean value : *real*8*

sigma- distribution standard deviation : *real*8*

sx- error on mean value : *real*8*

This routine computes the mean value of the x variable, its standard deviation an statistical error on the x variable, within the range set by **xmin** and **xmax**. The contents c_i of the histogram are used as the (non-normalized) probability distribution. The applicability conditions and used formulas are the same as in **ulhxmean**. This routine is particularly useful for peaks in the histogram.

The routine should be used only for histogram satisfying the following conditions:

All event have weight **wght=1**

All score quantities are equal to **1** (ie q=1)

Histogram is type **1**

****

ulhpeak(id,xmin,xmax,xmed,sigma,total,anet)

****

Purpose: Peak analysis

id- histogram id : *integer*4*

xmin -x lower value : *real*8*

xmax- x upper value : *real*8*

xmed- peak centroid : *real*8*

sigma- peak standard deviation : *real*8*

total- area under peak : *real*8*

anet- net area bgd subtracted : *real*8*

The analysis of the peak between **xmin** and **xmax** is made. The peak is assumed to over background events. The peak centroid, standard deviation, and net area are obtained after background subtraction. The number of background events is obtained interpolating a straight line between **xmin** and **xmax**. This value correspond to channel i_{min} and i_{max} with contents c_{imin} and c_{imax}, so the background values at each point x_i can be computed as:

$$\mathrm{bgd}_i = \mathrm{m} \times \mathrm{x}_i + \mathrm{b} \quad \text{where} \quad m = \frac{c_{imin} - c_{imax}}{x_{min} - x_{max}} \quad \text{and} \quad b = c_{imin} - m \times x_{min} \quad .$$

The total area under the peak is:

$$\mathbf{total} = \sum_{i_{min}}^{i_{max}} \mathrm{c}_i \quad ,$$

and the background subtracted variables are thus obtained as:

$$\mathbf{xmed} = \frac{\sum(\mathrm{x}_i(\mathrm{c}_i - \mathrm{bgd}_i))}{\mathbf{anet}}$$ where is $\mathbf{anet} = \sum_{i_{min}}^{i_{max}} (\mathrm{c}_i - \mathrm{bgd}_i)$ the sum of contents over all channels,

$$\mathbf{sigma} = \sqrt{\frac{\sum \mathrm{x}_i^2(\mathrm{c}_i - \mathrm{bgd}_i)}{\mathbf{anet}} - \mathbf{xmed^2}} \quad .$$

In the case $c_i - bgd_i < 0$ the difference is set to zero to avoid negative frequencies in the histogram.

The routine should be used only for histogram satisfying the following conditions:

All event have weight **wght=1**

All score quantities are equal to **1** (ie q=1)

Histogram is type **1**

****

ulhfwhm(id,fwhm)

****

Purpose: Compute fwhm

id- histogram id	: *integer*4*
fwhm -full width at half-maximum	: *real*8*

Computes the full width at half-maximum a peak distribution. The left and right half-width at half-maximum are computed relative to local maximum near the mean value.

****

ulhgetcx(id,x,c,e)

****

Purpose: Get content and error for a given channel in position x

id- histogram id : *integer*4*

x- position : *real*8*

c- channel content : *real*8*

e- error : *real*8*

This routine retrieves content and error of a channel given an x position.

****

ulhgetcxy(id,x,y,c,e)

****

Purpose: Get content and error for a given channel in position (x,y)

id- histogram id : *integer*4*

x- position : *real*8*

y- position : *real*8*

c- channel content : *real*8*

e- error : *real*8*

This routine retrieves content and error of a 2-d histogram channel given an (x,y) position.

****

ulhgetci(id,i,c,e)

****

Purpose: Get content and error for a given channel.

id- histogram id : *integer*4*

i- channel : *integer*4*

c- channel content : *real*8*

e- error : *real*8*

This routine retrieves content and error of channel i.

****

ulhgetcij(id,i,j,c,e)

****

Purpose: Get content and error for a given channel.

id- histogram id : *integer*4*

i- channel : *integer*4*

j- channel : *integer*4*

c- channel content : *real*8*

e- error : *real*8*

This routine retrieves content and error of channel (i,j) of a 2-d histogram.

****

ulhputci(id,i,c,e)

****

Purpose: Put content and error for a given channel.

id- histogram id : *integer*4*

i- channel : *integer*4*

c- channel content : *real*8*

e- error : *real*8*

This routine allows to directly change the content and error of channel i. The histogram status is changed to 3 by this routine. The routines **ulhadd1, ulhacc1** and **ulhave1** cannot be called after this routine.

****

ulhputcij(id,i,j,c,e)

****

Purpose: Put content and error for a given channel.

id- histogram id : *integer*4*

i- channel : *integer*4*

j- channel : *integer*4*

c- channel content : *real*8*

e- error : *real*8*

This routine allows to directly change the content and error of channel (i,j) of a 2-d histogram. The histogram status is changed to 3 by this routine. The routines **ulhadd2** , **ulhacc2** and **ulhave2** cannot be called after this routine.

****

ulhgeti(id,x,i)

****

Purpose: Get the channel i given a position x.

id- histogram id : _integer*4_

x- position : _real*8_

i- channel : _integer*4_

****

ulhgetij(id,x,y,i,j)

****

Purpose: Get the channel (i,j) given a position (x,y) of a 2-d histogram.

id- histogram id : _integer*4_

x- position : _real*8_

y- position : _real*8_

i- channel : _integer*4_

j- channel : _integer*4_

****

ulhgetx(id,i,x)

****

Purpose: Get position x for a given channel i.

id- histogram id : _integer*4_

i- channel : _integer*4_

x- position : _real*8_

****

ulhgetxy(id,i,j,x,y)

****

Purpose: Get position (x,y) for a given channel (i,j) in a 2-d histogram.

id- histogram id : _integer*4_

i- channel : _integer*4_

j- channel : _integer*4_

x- position : _real*8_

y- position : _real*8_

****

ulhputid(id,ip,itype)

****

Purpose: Put an id into the database

id- histogram id : *integer*4*

ip- histogram pointer : *integer*4*

itype- histogram type : *integer*4*

This routine adds the histogram **id** to the database and gets its pointer **ip**. The routine is called by the booking routines **ulhb1**, **ulhb2** and **ulhcopy**.

It is not intended to be used by the user.

****

ulhgetip(id,ip,itype)

****

Purpose: Get the ID pointer IP from database

id- histogram id : *integer*4*

ip- histogram pointer : *integer*4*

itype- histogram type : *integer*4*

For histogram **id**, this routine gets its pointer **ip** from the database.

It is not intended to be used by the user.

Random number generators

****

ulhrndi(id)

****

Purpose: Initializes histogram ID as a random number generator

id- histogram id : *integer*4*

The routine initializes histogram ID as a random number generator. After initialization each channel content c(j) is replace by the normalized integral between channel 1 and channel j, i.e.

$$c'(j)=\frac{\sum_{i=1}^{i=j} c(i)}{S} \quad \text{where} \quad S=\sum_{i=1}^{i=n} c(i) \quad .$$

****

ulhrnd1(id,x)

****

Purpose: To generate a random number according to the distribution in histogram ID.

id- histogram id : *integer*4*

x- random variable : *real*8*

The routine gives a number x distributed according to distribution stored in histogram ID. The inverse transformation method is used in this routine.

7. List of user routines

ulhacc1(id,x,q,iev)

ulhacc2(id,x,y,q,iev)

ulhadd1(id,x,q)

ulhadd2(id,x,y,q)

ulhave1(id,x,q)

ulhave2(id,x,y,q)

ulhb1(id,title,nx,xmin,xmax)

ulhb2(id,title,nx,xmin,xmax,ny,ymin,ymax)

ulhbf1(id,tit,nx,xmin,xmax,fun)

ulhclose

ulhcopy(id1,id2)

ulhdiv(id1,id2,id3,a)

ulhend(id)

ulhexp(id1,id3,b)

ulhfill1(id,x,q,wght,iev)

ulhfill2(id,x,y,q,wght,iev)

ulhfwhm(id,fwhm)

ulhgetci(id,i,c,e)

ulhgetcij(id,i,j,c,e)

ulhgetcx(id,x,c,e)

ulhgetcxy(id,x,y,c,e)

ulhgeti(id,x,i)

ulhgetij(id,x,y,i,j)

ulhgetx(id,i,x)

ulhgetxy(id,i,j,x,y)

ulhin1(id)

ulhin2(id)

ulhinfo(id,hinfo)

ulhinsp(id)

ulhlist(id)

ulhlog(id1,id3)

ulhmul(id1,id2,id3,a)

ulhopen(iunit,filename,utype)

ulhout(id)

ulhpeak(id,xmin,xmax,xmed,sigma,total,anet)

ulhputci(id,i,c,e)

ulhputcij(id,i,j,c,e)

ulhputid(id,ip,itype)

ulhreset(id)

ulhrnd1(id,x)

ulhrndi(id)

ulhscale(id1,id3,a)

ulhsetype(id,itype)

ulhsetitle(id,title)

ulhsqrt(id1,id3)

ulhstart

ulhsum (id1,id2,id3,a,b)

ulhxm(id,xmin,xmax,xmed,sigma,sx)

ulhxmean(id,xmed,sigma,sx)

Common variables

The following are the common parameters and variables. They can be found in file "**ulhistos.inc**". This file is automatically included in every routine where this variables are necessary. If the user requires a bigger number of histograms or number of channels the modification should be made in this file.

```
 character*5 version_ulhistos_inc
 parameter(version_ulhistos_inc='3.1')

parameter(nmax1d=10000) ! Max number of channels per 1-d histos
parameter(nmax2d=200)  ! Max number of channels is nmax2d**2 for 2-d histos
parameter(nh1d=100)    ! Max number of 1-d histos
parameter(nh2d=10)     ! Max number of 2-d histos
parameter(npoint=9999) ! ID's range = dimension of pointers array

common/ulhistos1/histo1d(nh1d,nmax1d),histo1dv(nh1d,nmax1d),
     &       nx1d(nh1d),
     &       xmin1d(nh1d),xmax1d(nh1d),binx1d(nh1d),
     &       underflow1d(nh1d),overflow1d(nh1d),
     &       nentries1d(nh1d),
     &       istatus1d(nh1d)
      common/ulhistos2/histo2d(nh2d,nmax2d,nmax2d),
     &       histo2dv(nh2d,nmax2d,nmax2d),
     &       nx2d(nh2d),ny2d(nh2d),
     &       xmin2d(nh2d),xmax2d(nh2d),binx2d(nh2d),
     &       ymin2d(nh2d),ymax2d(nh2d),biny2d(nh2d),
     &       underflow2d(nh2d),overflow2d(nh2d),
     &       nentries2d(nh2d),
     &       istatus2d(nh2d)

      common/ulhpartial1/nev1d(nh1d),histo1dp(nh1d,nmax1d)
      common/ulhpartial2/nev2d(nh2d),histo2dp(nh2d,nmax2d,nmax2d)

      common/ulhweight/wghtev1d(nh1d),wghtev2d(nh2d),
     &              wghtot1d(nh1d),wghtot2d(nh2d)
      common/ulhwweight2/iwghtf1(nh1d),iwghtf2(nh2d)    !flag

      common/ulhfrec/
     & histo1dw(nh1d,nmax1d),histo2dw(nh2d,nmax2d,nmax2d),
     & histo1de(nh1d,nmax1d),histo2de(nh2d,nmax2d,nmax2d)

c     iddim : dimension of histo id
c     idtyp : type of histo id
      common/ulhistoid/ids1d(nh1d),ids2d(nh2d),
     &               ids1d_1(npoint),ids2d_1(npoint),
     &               iddim(npoint),idtyp(npoint),
     &               nids,nids1d,nids2d
      common/units/iue,ius,iuo,iui

c
      character*64 title1d,title2d
```

```
 common/titles/title1d(nh1d),title2d(nh2d)

COMMON/ULHRSEED/IULHSEED1,IULHSEED2  ! random generator
```

Status flag

status=0 : histogram can be filled

status=1 : frequency type histogram. Histogram cannot be filled. Operations are allowed.

status=2 : event average or channel average type histogram. Histogram cannot be filled. Operations are allowed.

status=3 : contents and errors set by user. Histogram cannot be filled. Operations are allowed.

status=4: contents according to a given function

status=5: histogram used to generate a random variable

9. References

[1] HBOOK, CERN Program Library Long Writeup Y250

[2] GNUPLOT, http://www.gnuplot.info/

This text has been prepared using the open-source software Openoffice: http://www.openoffice.org