Electron-positron Annihilation

These analysis examples are taken from **examples/expl2.optns** (electron-positron annihilation). Please refer to the "Complete examples" section to get a complete description of the simulation configuration.

Multiplicity Distribution

```
!--------------------------------------------------
!     Define analysis (multiplicity distribution)
!--------------------------------------------------
beginanalysis
  histogram
    mulevt !x = multiplicity
    numevt !y = number of events
    1      !normalisation (per event)
    -0.5   !x-min
    61.5   !x-max
    31     !number of bins
  idcode  9970 !charged particles
endanalysis

!--------------------------------------------------
!      Write out final results to output file
!--------------------------------------------------
write "multiplicity distribution"
histoweight
writearray 3
```

Define analysis

We first define the *xvariable* as **mulevt** (mutiplicity) and *yvariable* as **numevt** (number of events). The following four numbers define: the normalisation code (***1*** means that we perform a normalisation, dividing by the number of events), the multiplicity range (from ***-0.5*** to ***61.5***), the number of bins (***31***).

The **idcode** command defines the particles of interest: ***9970*** means that we focus only on charged particles.

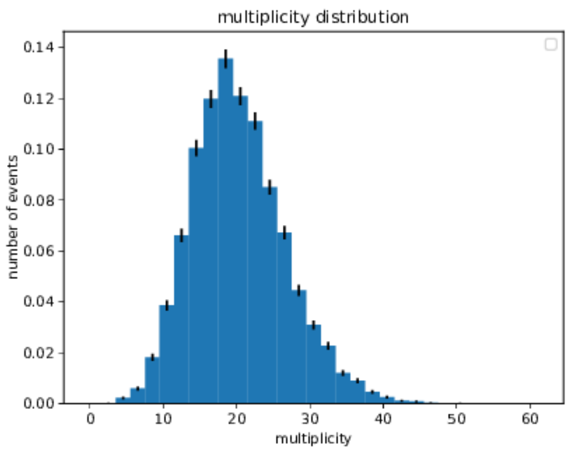
Output

The command **write** defines the histogram title (here: **multiplicity distribution**).

The command **histoweight** prints the histoweight value: here, the number of events triggered by the analysis.

Finally, the command **writearray *3*** creates a 3-column table with x, y and y-error values. The analysis results are written in the file **${HTO}z-expl2.histo**. We get 31 bins with mutiplicity values in the range]-0.5, 61.5[with a total of 10000 events.

One can then build and display the plot.



Rapidity Distribution

```
!---------------------------------------------------
!     Define analysis (rapidity distribution)
!---------------------------------------------------
beginanalysis
  histogram absrap numptl 11 0 6 30 !absrap = absolute value of rapidity
  frame thrust !particular frame used in e+e-
  idcode 9970
endanalysis

!--------------------------------------------------
!      Write out final results to output file
!--------------------------------------------------
write "rapidity distribution"
histoweight
writearray 3
```

Define analysis

We first define the *xvariable* as **absrap** (absolute value of rapidity) and *yvariable* as **numptl** (number of particles). The following four number define the normalisation code (***11*** means that we perform a normalisation dividing by the number of events and by the bin width), the absolute value of rapidity range (from ***0*** to ***6***), the number of bins (***30***).

The **idcode** commands define the particles of interest: ***9970*** means charged particles.

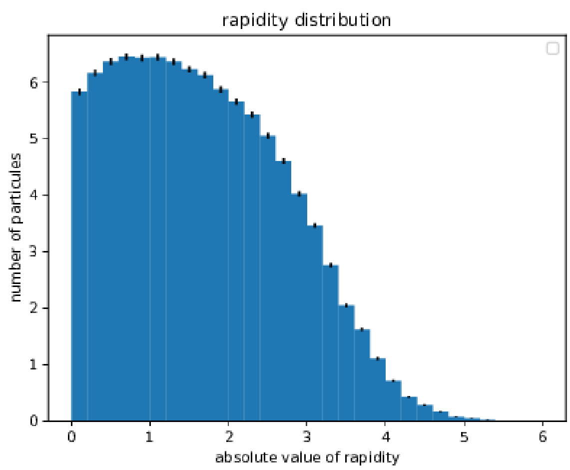
Output

The command **write** defines the histogram title (here: **multiplicity distribution**).

The command **histoweight** prints the histoweight value: here, the number of events triggered by the analysis.

Finally, the command ***writearray 3*** creates a 3-column table with x, y and y-error values. The analysis results are written in the file **${HTO}z-expl2.histo**. We get 30 bins with mutiplicity values in the range]0, 6[with a total of 10000 events.

One can then build and display the plots with one's own plotting tool. Here are the plot created with a simple python script using the matplotlib package.



XP Distribution

```
!---------------------------------------------------
!     Define analysis (xp distribution)
!---------------------------------------------------
beginanalysis
  frame total
  binning log
  histogram xp numptl 11 0.001 1 30
  idcode 9970
endanalysis

!---------------------------------------------------
!      Write out final results to output file
!---------------------------------------------------

write "xp distribution"
histoweight
writearray 3
```

Define analysis

We first define the *xvariable* as **xp** (x_p) and *yvariable* as **numptl** (number of particles). The following four numbers define the normalisation code (***11*** means that we perform a normalisation, dividing by the number of events and by the bin width), the xp range (from ***0.001*** to ***1***), the number of bins (***30***).

The ***idcode*** command defines the particles of interest: ***9970*** means charged particles.

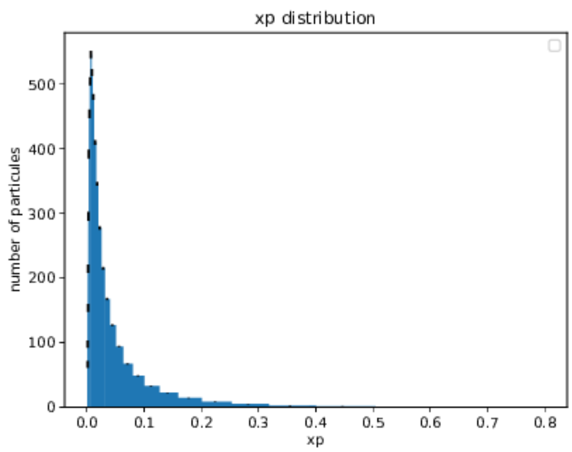
Output

The command **write** defines the histogram title (here: **xp distribution**).

The command **histoweight** prints the histoweight value: here, the number of events triggered by the analysis.

Finally, the command **writearray** ***3*** creates a 3-column table with x, y and y-error values. The analysis results are written in the file **${HTO}z-expl2.histo**. We get 30 bins with a number of particles in the range]0.001, 1[with a total of 10000 events.

One can then build and display the plot.



XI Distribution

```
!--------------------------------------------------
!     Define analysis (xi distribution)
!--------------------------------------------------
beginanalysis
  frame total
  binning lin
  histogram xi numptl 11 0.1 6 30
  idcode 9970
endanalysis

!--------------------------------------------------
!      Write out final results to output file
!--------------------------------------------------
write "xi distribution"
histoweight
writearray 3
```

Define analysis

We first define the *xvariable* as **xi** (ξ) and *yvariable* as **numptl** (number of particles). The following four numbers define the normalisation code (***11*** means that we perform a normalisation, dividing by the number of events and by the bin width), the xi range (from ***0.1*** to ***6***), the number of bins (***30***).

The **idcode** command defines the particles of interest: ***9970*** means charged particles.

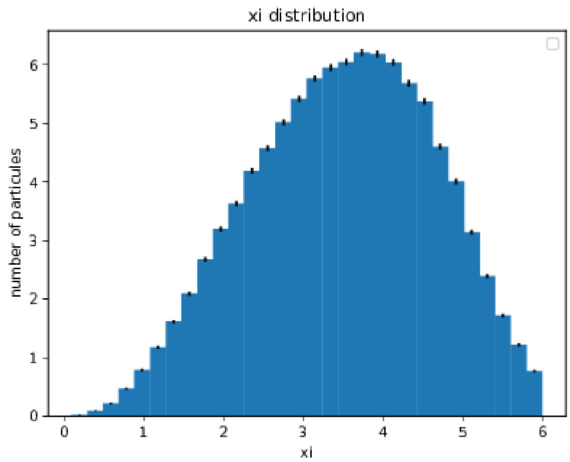
Output

The command **write** defines the histogram title (here: **xi distribution**).

The command **histoweight** prints the histoweight value: here, the number of events triggered by the analysis.

Finally, the command ***writearray 3*** creates a 3-column table with x, y and y-error values. The analysis results are written in the file **${HTO}z-expl2.histo**. We get 30 bins with a number of particles in the range]0.1, 6[with a total of 10000 events.

One can then build and display the plots with one's own plotting tool. Here are the plot created with a simple python script using the matplotlib package.



Ptr Distribution

```
!--------------------------------------------------
!     Define analysis (ptr-distr)
!--------------------------------------------------
beginanalysis
  frame thrust
  histogram p1a numptl 11   0.0001  10.0   20
  trigger rap   -10.0    10.0
  idcode      9970
endanalysis

!-------------------------------------------------
!      Write out final results to output file
!-------------------------------------------------
write "ptr distribution"
histoweight
writearray 3
```

Define analysis

We first define the *xvariable* as **p1a** ($|p_x|$) and *yvariable* as **numptl** (number of particles). The following four numbers define the normalisation code (***11*** means that we perform a normalisation, dividing by the number of events and by the bin width), the **p1a** range (from ***0.0001*** to ***10.0***), the number of bins (***20***).

The **idcode** command defines the particles of interest: ***9970*** means charged particles.

Output

The command **write** defines the histogram title (here: ***ptr distribution***).

The command **histoweight** prints the histoweight value: here, the number of events triggered by the analysis.

Finally, the command **writearray** ***3*** creates a 3-column table with x, y and y-error values. The analysis results are written in the file **${HTO}z-expl2.histo**. We get 20 bins with a number of particles in the range]0.0001, 10.0[with a total of 10000 events.

One can then build and display the plot.