The HERA II transverse polarimeter: analysis software and data access, Version 0

Stefan Schmitt

July 24, 2008

1 Introduction

1.1 Principle of measurement

The HERA lepton beam has a natural transverse polarisation from the Sokolov-Ternov effect. However, because there are depolarizing effects from imperfections of the HERA machine and from beam-beam interactions, the polarisation has to be monitored throughout each HERA fill. Furthermore, the polarisation may be significantly different for colliding and non-colliding bunches. It is the purpose of the transverse polarimeter to constantly measure the HERA beam polarisation.

Compton photons are scattered off the HERA lepton beam and detected in a small sampling calorimeter. The scintillator plates are arranged in two optically seperated halves, "up" and "down". The calorimeter is read out by 4 photo-multipliers. The U and D channels each are receiving light only from the upper or lower half of the calorimeter, respectively. The L and R channels are receiving light from both halves of the calorimeter. The quantity $\eta = \frac{U-D}{U+D}$ is sensitive to the vertical position of the incoming photon y, whereas $U + D$, L, R are sensitive to its energy.

The data are collected at a rate of typically 50 KHz and stored into histograms. The maximum energy of the Compton Photons is $\approx$ 14 GeV. Because there is non-negligable background from Bremsstrahlung, it is necessary to control it. For this reason a chopper is inserted in the laser beam, and background-only data are collected. Typical measurement coditions are: $40s$ of laser-on data alternated with $20s$ of laser-off data.

The Compton cross-section is sensitive to the lepton beam polarisation only if the laser is circular polarized. For this reason the laser is operated in alternating helicity states $S_3^{L,R} = \pm 1$. The helicity flip is done using a Pockel's cell at a rate of $\approx$ 80 Hz. The transverse beam polarisation leads to an asymmetric cross-section profile when projected on the y axis, and this asymmery changes sign when the sign of S_3 is flipped. The beam polarisation is proportional to the difference $\langle y \rangle^L - \langle y \rangle^R$, and hence

$$P = Ap \times \left(\langle \eta \rangle^L - \langle \eta \rangle^R \right). \tag{1}$$

However, because the asymmetry η has nonlinear dependence on the vertical position and because the resolution in η depends on the vertical position, the analysing power Ap is not a constant. It depends on the detector and on various beam parameters. A silicon device and a converter have been added in front of the TPOL detector in order to allow for a simultaneous measurement of y and η for converted photons. The readout of the silicon detector was not fast enought to provide polarisation numbers. Instead, special runs were collected with the silicon detector, and that data can be used to better understand the transformation $y \rightarrow y$ and eventually the analyzing power Ap.

1.2 TPOL data and its formats

Because the TPOL delivers rather high event rates, the calorimeter data are stored in histograms, rather than event-by-event. The histograms are ordered by their time-stamp.

In contrast, data from testbeams, from special runs with the silicon detector, and from the GEANT simulation are stored in event-by-event ntuples.

Data from the histograms are analyzed and the results (polarisation and other quantities) are kept in ntuples, where one "event" corresponds approximately to a distinct time-stamp, i.e. one minute of data taking. This is the main TPOL analysis step. The key problem in improving the TPOL analysis is to feed the knowledge acquired with the event-by-event data back into this analysis step.

The various types of data are summarized in table 1.

Source	Output type	Name	Comment
Calorimeter	packed Histograms	`*.rdata`	Input to `readraw`, `run2nt`
`readraw`	HBOOK Histograms	`*.rz`	monitoring, tests
`run2nt`	time ordered HBOOK Ntuple	`*.rz`	main results
Testbeam	packed event data	`*.data`	Input to `adc2nt`
Silicon and Calo	packed event data	`*.comb`	Input to `adc2nt`
`adc2nt`	event-by-event HBOOK Ntuple	`*.rz`	tune MC
GEANT MC	event-by-event HBOOK Ntuple	`*.rz`	tune MC

Table 1: TPOL data formats.

1.3 Existing TPOL analyses: an overview

1.3.1 The online analysis

The online analysis is based on the mean and RMS of the η distribution in selected energy bins. More details are givenm in section 6.4. The polarisation is calculated as specified in equation 1. With this approach the key problem is to understand the analyzing power Ap. At present the analyzing power is assumed to depend quadratically on the focus f. Other dependencies are neglected. The focus f is a measure of the vertical beam size at the calorimeter surface. It is calculated from the RMS of the η distribution in an energy bin where the beam size is expected to be dominated by the electron beam emittance. The dependence of the analyzing power on f has been studied using the so-called fast Monte Carlo. This program is not part of the official TPOL software. For the normalisation of the quadratic function derived this way, the fast Monte Carlo was rescaled to match the HERA I analyzing power. The HERA I analyzing power has been determied from risetime measurements.

In summary, the function Ap whcih is in use now is not well defined, and it can not be reproduced easily. For this reason, the present online analysis has rather large uncertainties on the absolute scale of the polarisation. On the other hand, the online analysis provides a robust and simple method for calculating polarisation numbers. All HERA II data can be processed on the timescale of approximately one week. Moreover, all relevant parameters extracted from the raw data can be stored in ntuples, so corrections to Ap can be tested without having to read the raw data again and again.

1.3.2 Multi-Parameter fits

Another ansatz which has been studied are multi-parameter fits. The basic idea is to start with the analytic Compton cross-section, smear it using the beam parameters and calorimeter resolution, then compare it to the background-subtracted data. The underlying parameters are tuned, and the χ^2 is minimized. One of the parameters is the beam polarisation.

Two alternative fit methods have been investigated. The original method is based on a staged fitting procedure and MINUIT. Three seperate fits are performed, and each of them gives a subset of the parameters. The more recent method involves fitting all parameters in one step, using fitting code different from MINUIT.

One potential advantage of this method is that the $\eta \to y$ transformation, measured with the silicon detector, can be added to the parametrisation in exact form. For the traditional "online" method, the $\eta \to y$ transformation does not enter directly. Instead, the Monte Carlo has to be tuned until it reproduces the measured function.

In summary, the multi-parameter fits have not been successful. The $\frac{\chi^2}{N_{\mathrm{DF}}}$ is significantly larger than 1, which is a hint that the parametric models are incomplete, despite of a very large number of parameters (≈ 30). Further increasing the number of parameters spoils the fit convergence or results in parameters outside the physical range, instead of improving the fit quality. Moreover, such fits are very slow. It requires several minutes of CPU time to fit 1 minute of data, because the processing time for evaluating χ^2 requires numerical integration for each point in the E, η histogram, and hence it is proportional to N^3 (N^4 for pileup effects), where N is the typical number of bins in E, η or y. For serious testing, several 100 minutes of data have to be processed, which significantly slows down the turn-around for improving the analysis. Running over all HERA data would require massive usage of parallel computing (e.g. GRID).

2 Location of the polariameter data

2.1 Analysis Software

The TPOL analysis software is stored on afs. A copy of the software can be obtained using CVS:

cvs -d /afs/desy.de/user/p/pol2000/CVSroot co daq

Note that You have to become a member of the afs group **pol2000:tpol** in order to access the software. For inspecting the group members, type

pts membership pol2000:tpol

. The "daq" package includes software to access the data offline, as well as all the program which have been used during data taking.

Another package contains a GEANT3 simulation of the TPOL.

cvs -d /afs/desy.de/user/p/pol2000/CVSroot co tpolmc

2.2 CVS tags

Up to now, CVS tags have not been used in a systematic manner.

2.3 Data storage

The TPOL data is stored in the DESY disk-cache. The data are accessible by the disk-cache software (like *dccp* to read files). The TPOL software is prepared to read directly from the disk-cache, using the *dcap* library. Within the H1 environment the diskcache is mounted as **/acs** and all TPOL data are stored in **/acs/tpol**. The remote PNFS directory is `pnfs.desy.de:/tpol/public`.

/acs/tpol/02/hera_testbeam testbeam data collected at DESY

/acs/tpol/02/cerntest testbeam data collected at CERN

/acs/tpol/02/testbeam selected testbeam data in HBOOK format, packed into `*.tar` files

/acs/tpol/02 data collected in 2002

/acs/tpol/03 data collected in 2003

/acs/tpol/04 data collected in 2004

/acs/tpol/05 data collected in 2005

/acs/tpol/06 data collected in 2006

/acs/tpol/07 data collected in 2007

/acs/tpol/backup/database slow control information

/acs/tpol/backup/repro_2007_06_07 results from rerunning the online-type analysis in Summmer 2007, in HBOOK format

/afs/desy.de/user/p/pol2000/public/ntuples results from combining the TPOL and LPOL data and averaging over periods of 1 hour, in HBOOK format. There is a file **rationt.txt**, giving a short description of the variables. There is one HBOOK file per year of data taking. The ntuples reflect the status in Summer 2007.

Within each directory **/acs/tpol/YY** where **YY** is the year, there is the following structure:

/acs/tpol/YY/Jan Data files for each run, collected in January.

. . .

/acs/tpol/YY/Dec Data files for each run, collected in December.

/acs/tpol/YY/fills Data files for each fill collected in that year.

For standard analyses there is no need to access the raw data files written per run in the month-by-month directories. It is recommended to read only the files from the **fills** directory, because disk-cache usage is more efficienct when reading one large file instead of many small files.

The following types of files are available for analysis

/acs/tpol/YY/fills/FILLffff_sssss_eeeee.rdata Calorimeter data (histograms) collected in the year **YY**, with HERA fill number **ffff**, including the TPOL runs **sssss** up to **eeeee**.

/acs/tpol/YY/MMM/RUNrrrrr.rdata Calorimeter data (histograms) collected in the year **YY** and month **MMM**, with TPOL run number **rrrrr**. *If large amount of data have to be read, please use the FILL files instead.*

/acs/tpol/YY/MMM/RUNrrrrrlp.tgz Light polarisation measurement collected in the year **YY** and month **MMM** between HERA fills, with TPOL run number **rrrrr**.

/acs/tpol/YY/MMM/tpol.rrrrr.eeeee.comb Calorimeter and silicon detector data (event by event), collected in the year **YY** and month **MMM**, in parallel to the TPOL run number **rrrrr**, with event data run number **eeeee**.

/acs/tpol/YY/MMM/tpol.rrrrr.eeeee.calo same as above, but the silicon detector has not been read out

/acs/tpol/YY/MMM/tpol.rrrrr.eeeee.silicon same as above, but the calorimeter has not been read out

/acs/tpol/YY/MMM/comb.eeeee.rz Calorimeter and silicon detector data (event by event), collected in the year **YY** and month **MMM**, with event data run number **eeeee**, in HBOOK ntuple format.

3 TPOL data in HBOOK format

As said in table 1, there are three types of HBOOK files produced by the TPOL software. However, only two of these are stored permanently on disk. In particular, the raw data histograms are not stored in HBOOK format. They can be retreived on demand, using the "readraw" program, as explained in the examples in section 5.

3.1 Content of the time ordered HBOOK ntuple

The ntuples stored in **/acs/tpol/backup/repro_2007_06_07** have been obtained from the TPOL raw data by rerunning the (then) latest version of the online analysis in summer 2007, using the **run2nt** program. The most relevant variables of these ntuples are summarized in table 2.

Calculated from Histograms	
p_tpol ± dp_tpol	TPOL polarisation all bunches
pnc_tpol ± dpnc_tpol	TPOL polarisation non-colliding bunches
pc_tpol ± dpc_tpol	TPOL polarisation colliding bunches
spot ± dspot	Compton cone offset all bunches
spotnc ± dspotnc	Compton cone offset non-colliding bunches
spotc ± dspotc	Compton cone offset colliding bunches
focus ± dfocus	Compton cone size all bunches
focusnc	Compton cone size non-colliding bunches
focusc	Compton cone size colliding bunches
ds1 ± dds1	difference of laser polarisation $S_L^1 - S_R^1$
lumi ± dlumi	Compton luminosity
distip ± ddistip	Distance Compton IP to calorimeter
elow	Average energy used for distance measurement
Timing and normalisation	
tpoltime	unix time-stamp
offrate	trigger rate laser off
onrate	trigger rate laser on
offdead	deadtime laser off
ondead	deadtime laser on
Gain calibration	
alr±dalr, salr±dsalr	mean and width of $\frac{L}{L+R}$
elr±delr, selr±dselr	mean and width of $L+R$ Compton edge
eud±deud, seud±dseud	mean and width of $U+D$ Compton edge
fu±dfu, fd±dfd	ratio $\frac{U+D}{L+R}$ extrapolated to $D=0$, $U=0$
fdown, fup	Calibration constants actually used
Slow control	
x_table, y_table	calorimeter table position
m3h, m3v	mirror position M3
hvup,hvdown,hvleft,hvright	high voltage settings
run,cycle	TPOL run number and cycle number
s1neg_abox±ds1neg_abox	latest optical meas., neg. PC HV $\sqrt{S_1^2+S_2^2}$
s1pos_abox±ds1pos_abox	latest optical meas., pos. PC HV $\sqrt{S_1^2+S_2^2}$
fill,energy,current,protcur	HERA machine parameters
x_herab,y_herab,x_wr133,y_wr133	HERA BPMs closest to TPOL

Table 2: Most relevant variables stored in the ntuple produced by the run2nt program.

3.2 Event-by-event HBOOK Ntuples

Event-by-event data ntuples are available from the CERN testbeam and from silicon runs taken during HERA operation.

CERN testbeam get a copy of the file `/acs/tpol/02/testbeam/cerntest.tar` and unpack it:

```
mkdir cerntest
```

```
cd cerntest
dccp /acs/tpol/02/testbeam/cerntest.tar cerntest.tar
tar xvf cerntest.tar
```

A text file **cerntest_ntuple.txt** describing the ntuple is included. The ntuple files are named **cerntest_eE_rR.rz**, where $E = [06, 08, 10, 12.5, 15, 17.5, 20, 22.5, 25, 27.5, 30, 40, 50]$ is the beam energy in GeV, and R is the first run number. For convenience, several runs with the same beam energy have been merged into one file. For an energy of 10 GeV, two ntuples with a total number of ≈ 900000 events are available. For the other energies, the typical number of events is $50000 - 100000$, stored in a single ntuple.

HERA data the HERA event-by-event ntuples are stored under `/acs/tpol/YY/MMM/*.rz`, where YY is the year and MMM is the month. It is probably best to contact Blanka Sobloher about a list of the most useful runs and which quality cuts to apply. The Ntuple variables are similar to those from the testbeam. The main differences are related to the input-output register and timing corrections. Timimg corrections are necessary for the testbeam data only, because the testbeam clock phase was not locked to the sampling ADC clock.

The most relevant variables are summarized in table 3.

Variable	testbeam	HERA
energy	Beam energy	not stored
tablex	Table x position	not stored
tabley	Table y position	not stored
tdc	trigger-clock phase	not stored
ioreg	finger counter	chopper and helicity
elr	$L + R$ energy with timing corr.	L+R energy
eud	$U + D$ energy with timing corr.	U+D energy
alr	$\frac{L-R}{L+R}$ with timing corr.	$\frac{L-R}{L+R}$
eud	$\frac{U-D}{U+D}$ with timing corr.	$\frac{U-D}{U+D}$
ncy	number of silicon y clusters	
cypos(ncy)	silicon cluster position	
cychg(ncy)	silicon cluster charge	
cyrad(ncy)	silicon cluster radius	

Table 3: The most relevant variables of the event-by-event ntuples.

4 Compiling the TPOL software

4.1 GEANT Monte Carlo

After checking out the software (section 2), go to the directory **tpolmc** and type **make**.

4.2 TPOL analysis software

Make sure the environment variable **OSTYPE** is set. For ksh, zsh, bash, type *export OSTYPE=linux*. After checking out the software (section 2), go to the directory `daq` and type **make**. This will set up the dependencies of all libraries and executables. The simple "make" has to be repeated each time new source files are included or if the dependencies of header and source file change. To compile a program, type **make linux/PROG**, where **PROG** is the name of the program.

5 TPOL analysis software structure

5.1 Directory structure

daq/include header files of the tpol software

daq/src source files of the tpol software

daq/makefile main TPOL makefile

daq/makefile.paths makefile with package names and paths

daq/makefile.libs makefile with library and executables specification.

daq/makefile.flags makefile with platform-independent flags

daq/makefile.linux makefile with linux-specific flags

daq/makefile.tpol automatically generated makefile containing all source-file dependencies. This file is created by typing **make**. If it is corrupted, remove it, create an empty file, and type make again.

5.2 Packages

The sourcefiles and header files are organized in packages.

daq/include/PACKAGE header files

daq/src/PACKAGE source files

All source files from one or more packages are compiled into a common library. The relation between package and library name is given in the files **makefile.libs** and **makefile.paths**. Only a subset of the packages, relevant for the online-type analysis, is described here.

- **linux/libtpol.a:** general TPOL library, contains files from

  src/data Code related to the representation of the histograms in memory and on disk.

  src/runstatus Code related to the representation of the slow control information in memory and on disk.

- **linux/libtpolana.a:** TPOL analysis library, contains files from

  src/analysis Online-type analysis code, multi-parameter fit, hbook interface.

5.2.1 Executables

All executable programs, containing a *main()* function are located in **src/executables**. Here only those executables are listed which are more or less stable and useful for offline analyses.

src/executables/tpolmake.C something similar to the lunix tool **makedepend**. This program is compiled and executed with proper arguments when typing **make** without arguments.

src/executables/readraw.C read raw data (histograms) and write out HBOOK histograms.

src/executables/run2nt.C read raw data (histograms), rerun the online analysis and write out HBOOK ntuples.

src/executables/adc2nt.C read (silicon and/or calo) event data files and write out HBOOK ntuples.

Below is a list of programs in experimental status. For development only!!!

testff run the multi-parameter fit.

toy simulate toy Monte Carlo using the fit parameters.

5.3 Examples for reading the TPOL raw (histogram) data

Compile the software to read TPOL raw data files:

1. cvs -d /afs/desy.de/user/p/pol2000/CVSroot co daq
2. cd daq
3. export OSTYPE=linux
4. make linux/readraw
5. make linux/run2nt

Example: convert the 10 first cycles of raw data from fill 4860 to HBOOK format and rerun the online analysis for all cycles in fill 4860:

1. linux/readraw -n 10 -r /acs/tpol/07/fills/FILL4860_77700_77861.rdata -o fill4860.rz
2. linux/run2nt -F -d /acs/tpol/07/fills FILL4860_77700_77861.rdata -o fill4860ana.rz

The output file "fill4860.rz" contains 10 sets of histograms with TPOL raw data. The histogram sets have an offset of 2000 per set. See **kumac/monitor.kumac** how to plot some basic histograms for the first pair of laser on/off data in a file named "test.rz".

The output-file "fill4860ana.rz" contains a ntuple with the analysis results of all cycles in fill 4860. See table 2 for more details about its content.

Please note that the programs may appear to be hanging, while they are waiting for disk-cache access. Just be patient.

6 Some details about the TPOL C++ code

6.1 The Histogram and Histogram1 class

Raw data histograms are stored using dedicated histogramming code, stored in *histogram.h*. One-dimensional histograms are stored in the template class **Histogram1**. Two-dimensional histograms are stored in the template class **Histogram**. Only aequidistant bins are supported. Overflows are not kept. The most relevant methods are summarized in table 4.

6.2 The RawData class

The smallest unit of TPOL raw data is called a "cycle". A cycle consists of a set of histograms, scaler readings, time stamps and other slow control information. The histograms have been filled with Compton or background events, depending on the laser status. A single cycle is stored in the template class **RawData**. The derived classes **RawDataMem** and **RawDataMemConst** are used inside the analysis software. The data members are summarized in table 5 and in table 6. The relevant header files are *rawdata.h* and *rawdatamen.h*.

methods	description
x0(), x1(), y0(),x1()	upper/lower edge of histogram
nx(), ny()	number of bins
xi(i), yj(j)	bin center of bins i, j
Reference to bin content for 2-dimensional Histograms	
[uj][ui]	unsigned indices for access by bin number
[y][x]	float indices for access by x,y position
Reference to bin content for 1-dimensional Histograms	
[ui]	unsigned indices for access by bin number
[x]	float indices for access by x position

Table 4: Methods available for the histogram classes.

HBOOK id	RawData data member	Comment
	TpolHset StdHist[2][2]	histograms for analysis
$1000 + h + 2b$	Histogram<Short> Naud_eud_b[220][2]	single bunch histograms
$140 + c$	Histogram1<Integer> Nped[4],Ncog[4]	histograms for monitoring
$41, 42$	Histogram1<Integer> TriggerBits,Bunch	histograms for monitoring
$70 - 72$	Histogram1<Integer> LRped,LRchg,LRcog	histograms for monitoring
	Trigger trigger	timing information
	ScalerRate rate	scaler readings
	RunStatus sstart,sstop	slow control information
	GlobalTime tstart,tstop	unix time stamps

Table 5: Data members of the class RawData. The first index of StdHist[2][2] is 0 for non-colliding bunches and 1 for colliding bunches. The second index serves to address the two laser helicity states. The id is used inside HBOOK.

HBOOK id	RawData::TpolHset data member	Comment
$100 + h + 2b + 4c$	Histogram1<Integer> Nraw[4]	energy spectra of U, D, L, R
$200 + h + 2b$	Histogram1<Integer> Neud	energy spectrum $U + D$
$210 + h + 2b$	Histogram1<Integer> Nelr	energy spectrum $L + R$
$230 + h + 2b$	Histogram1<Integer> Nalr	distribution $\frac{L}{L+R}$
$320 + h + 2b + 4w$	Histogram1<Stat> Sr_aud	histogram of $\frac{U}{U+D}$ weighted by $\frac{U+D}{L+R}$
$240 + h + 2b$	Histogram<Integer> Naud_eud	2-dim histogram $U + D$ wrt $\frac{U}{U+D}$

Table 6: Data members of the class RawData::TpolHset. Weighted histograms are holding three seperate spectra: weighted by 1, weighted by w and weighted by w^2.

6.3 The RunFile class

For reading **RawDataMem** objects one can use the class **RunFile**, header file *runfile.h*. A RunFile consists of a sequence of cycles. The **read** method may be used to fill an existing **RawDataMem** with data from one of the numbered cycles in the file.

Example: read cycle n=500 from a data file and count the number of entries in the Naud_eud histograms.

```
#include "runfile.h"
#include "rawdatamem.h"

int main(int argc, char *argv[]) {
  RawDataMem cycle("Naud_eud Nelr Nalr Neud Sr_aud");
  RunFile file("/acs/tpol/07/fills/FILL4860_77700_77861.rdata");
  int n=500;
  file.read(cycle,n);
  for(int i=0;i<2;i++)
    for(int j=0;j<2;j++) {
      int nevent=0;
      RawDataMem::TpolHset const &h=cycle.StdHist[i][j];
      for(unsigned k=0;k<h.Naud_eud.nx();k++) {
        for(unsigned l=0;l<h.Naud_eud.ny();l++) {
          nevent += h.Naud_eud[l][k];
        }
      }
      cout<<"nevent["<<i<<"]["<<j<<"]="<<nevent<<"\n";
    }
  return 0;
}
```

```
nevent[0][0]=14705
nevent[0][1]=14734
nevent[1][0]=350818
nevent[1][1]=347864
```

Here are instructions how to compile this example program in the TPOL environment:

1. copy the program text into a file **src/executables/test1.C**
2. add the following line to the file **makefile.libs**
   # tpolexe test1 LIB LIBAN -$(LIBDC)
3. type **make** to update the file **makefile.tpol**
4. type **make linux/test1** to compile the program **linux/test1**
5. type **linux/test1** to run the program

6.4 The online analysis

The online analysis is coded in class **CALFITStatusDataW** and **POLFITStatusDataW**. The code is located in the files *dataana.h* and *dataana.C*, the real work is done in *analysis.h* and *analysis.C*. The calculation of additional quantities, based on these online algorithms have been added in a seperate class **DataAnalysis**, located in the file *dataana2.h*. The analysis is performed in two steps. The second step depends on the gain factors derived in the first step. Each step has to be called twice, once with laser-on and once with laser-off data in order to get results. Analysis steps:

1. using methods from class **CALFITStatusDataW** the gain calibration is performed
2. in class **POLFITStatusDataW** the 2-dimensional histograms of $E = U + D$ and $\frac{\eta+1}{2} = \frac{U}{U+D}$ are analyzed using the gain factors. Several quanitites, like the online polarisation, are calculated. Additional quantities are calculated in class **DataAnalysis**.

6.4.1 Background normalisation

The data and background histograms are normalized, using the total measurement time. Deadtime effects are corrected by comparing the observed event rate with independent scaler readings.

6.4.2 The gain calibration (analysis step 1)

The gain factors for the left and right channel are derived from histograms of $L + R$ and $\frac{L}{L+R}$. A Gaussian is fitted to the $\frac{L}{L+R}$ histogram, with mean μ_{alr}. The $L+R$ histogram is differentiated and a Gaussian is fitted to the Compton edge near 14 GeV, with mean μ_{elr}. The L, R gain factors are calculated as

$$g_L = \frac{E_0}{2\mu_{elr}\mu_{alr}},\ g_R = \frac{E_0}{2\mu_{elr}(1-\mu_{alr})}$$

where E_0 is the expected Compton energy. For calibrating the U and D channels the $U + D$ histogram and a profile plot of $R = \frac{U+D}{L+R}$ as a function of $\frac{U}{U+D}$ is used. The $U + D$ histogram is differentiated and the Compton edge is fitted, with mean μ_{eud}. The R profile plot is fitted by a parabolic function, with function value f_u for $\frac{U}{U+D} = 1$ and f_d for $\frac{U}{U+D} = 0$. The gain factors are calculated as

$$g_U = \frac{E_0(f_u + f_d)}{2\mu_{eud}f_u},\ g_D = \frac{E_0(f_u + f_d)}{2\mu_{eud}f_d}.$$

Under normal conditions the gain factors were kept at 1 within 0.5% by adjusting the photomultiplier high voltage. Note that all histograms are background-subtracted prior to analysing. The colliding/non colliding bunches and the two laser helicities are averaged.

6.4.3 The cross-section analysis (analysis step 2)

The two-dimensional histogram in $E = U + D$ and $\frac{\eta+1}{2} = \frac{U}{U+D}$ is analyzed in this step. The $(E_{\rm cal}, \eta_{\rm cal})$ plane is divided into six areas, where the index cal indicates that the gain factors have been applied. The areas are defined in table 7. For each area A_i the number of events is counted,

name	$E_{\rm cal}$ [GeV]	$\eta_{\rm cal}$	derived quantity
A_0	$0. < E < 5.225$	$\frac{-57}{64} < \eta < \frac{57}{64}$	IP distance D_{IP}, energy $E_{\rm low}$
A_1	$5.225 < E < 11.4$	$\frac{-57}{64} < \eta < \frac{57}{64}$	polarisation P
A_2	$11.4 < E < 13.775$	$\frac{-57}{64} < \eta < \frac{57}{64}$	beam spot y_0, focus σ_y
A_3	$13.775 < E < 16.15$	$\frac{-57}{64} < \eta < \frac{57}{64}$	unused
A_4	$16.15 < E < 30.4$	$\frac{-57}{64} < \eta < \frac{57}{64}$	unused
A_5	$5.225 < E < 11.4$	$\frac{-7}{32} < \eta < \frac{7}{32}$	linear light ΔS_1
A_6	$5.225 < E < 30.4$	no cut	luminosity L

Table 7: areas defined in the $(E_{\rm cal}, \eta_{\rm cal})$ plane.

weighted by 1, E, η, η^2. For this calculation the gain factors are taken into account by linear resampling of the original histogram: if a bin of the original histigram is fully contained in the area defined by the calibrated quantities, all events are counted, using the centre-of-gravity of the bin in

$(E_{\rm cal}, \eta_{\rm cal})$. Otherwise, the bin is replaced by some polygonial shape, such that its corners fit into the area defined by the calibrated quantities. A fraction of the events, proportional to the area of the polygonial shape is counted, using the new centre-of-gravity. These calculations are carried out seperately for laser-on, laser-off data, both helicities, and colliding/non-colliding bunches (a total of eight combinations). Using those numbers it is possible to calculate the MEAN and RMS of η and similar quantities, with background subtraction and gain correction applied. The following quantities are derived:

$$D_{\rm IP} \propto {\rm RMS}(\eta(A_0)),$$
$$E_{\rm low} \propto {\rm MEAN}(E(A_0)),$$
$$y_0 \propto {\rm MEAN}(\eta(A_2)),$$
$$\sigma_y \propto {\rm RMS}(\eta(A_2)),$$
$$\Delta S_1 \propto \frac{N(A_5^L) - N(A_5^R)}{N(A_5^L) + N(A_5^R)},$$
$$L \propto N(A_6),$$
$$P = Ap(\sigma_y) \times ({\rm MEAN}(\eta(A_1^R)) - {\rm MEAN}(\eta(A_1^L))).$$

The analyzing power is calculated as a quadratic function of the focus σ_y.

7 Do do list

7.1 GEANT Monte Carlo

- Improved steering to simulate Testbeam or Compton Monte Carlo
- Accurate HERA beam line simulation based on machine files
- GEANT description of photon beam transport system, aperture limits, etc
- Bremsstrahlung background
- Pileup effects
- Simulate Digitisation effects (ADC, pedestal subtraction, etc)
- Tools to reweight MC events for different beam parameters, polarisation, gain factors, etc and produce
  1. event-by-event ntuples, to be compared directly to testbeam and/or HERA ntuples
  2. raw data histograms, to be fed through the polarisation analysis

7.2 Polarisation analysis

- Tune GEANT MC to match testbeam and/or silicon data
- Generate "raw data histograms" for many parameters and quantify the impact on the quantities which can be measured (i.e. the results of the online analysis)
- Derive new analyzing power, based on measured quantities
- Compare to LPOL, identify periods with technical problems of either polarimeter, etc