UNIVERSITY
of
GLASGOW

Department of Physics & Astronomy
Experimental Particle Physics Group
Kelvin Building, University of Glasgow,
Glasgow, G12 8QQ, Scotland
Telephone: +44 (0)141 339 8855 Fax: +44 (0)141 330 5881

GLAS-PPE/2003-20
17th December 2003

Fitting Program for HPD Backpulse Spectra

A.Pickford[1]
[1] University of Glasgow, Glasgow, G12 8QQ, Scotland

Abstract

A program for fitting photoelectron spectra from HPDs is presented. The program uses a Fourier Transform method to give fast, efficient performance. The HPD response model implemented includes both the fully absorbed and backscattered photoelectrons. Noise can be included both at the single photoelectron response level and at the global detector response level, either as measured noise functions or Gaussian distributions.

1 Introduction

Hybrid photon detectors [1] (HPDs) are to be used as the photon detector for the LHCb RICH [2] detectors. A schematic of a HPD is shown in figure 1. A photoelectron released by an incident photon is accelerated from the photo-cathode through 20 kV on to a silicon detector. The silicon detector comprises a 300μm thick pixel array, bump bonded to a binary readout electronics chip. For calibration and testing the total analogue signal from the silicon detector as a whole (the backpulse signal) is available.

The computer code described in this paper has been used to fit the backpulse signal to a convolution of a single photoelectron response function and a Poisson distribution of the number of photoelectrons.

2 Fitting Theory

The method used in the code to fit the backpulse spectra is the 'light spectra sum rule' [3]. When the photon production rate is constant and photoelectrons are produced independently of one another then the number of photoelectrons produced follows Poisson statistics.

The response of the detector to a single photoelectron is described by a probability density function (p.d.f.) $r(x;\underline{a})$ which gives the probability of a single photoelectron giving a signal x, where $\underline{a}$ is a vector of parameters describing the p.d.f. to be determined by the fitting.

The signal from n photoelectrons is then the sum of the signals from each individual photoelectron. The p.d.f. for this signal is then the n-fold self convolution of the single photoelectron p.d.f.

$$r^{*n}(x;\underline{a}) = \int_0^\infty r^{*(n-1)}(y;\underline{a})r(x-y;\underline{a})dy \tag{1}$$

The p.d.f. for observing a signal, x and n photoelectrons, where the photoelectron production follows a Poisson distribution with mean μ is

$$s(x,n;\mu,\underline{a}) = e^{-\mu}\frac{\mu^n}{n!}r^{*n}(x;\underline{a}) \tag{2}$$

To include the effects of the readout chain $s(x,n;\mu,\underline{a})$ is then convoluted with the detector response function p.d.f. $g(x;\underline{b})$.

The p.d.f. for observing a signal x from a mean number of photoelectrons μ is then

$$s(x;\mu,\underline{a},\underline{b}) = \left(\sum_{n=0}^{\infty} e^{-\mu}\frac{\mu^n}{n!}r^{*n}(x;\underline{a})\right) \times g(x;\underline{b}) \tag{3}$$

where $\times$ indicates convolution. Using equation 3 when fitting for μ, $\underline{a}$ and $\underline{b}$ requires the calculation of an infinite sum. This is potentially time consuming and requires truncating the sum.

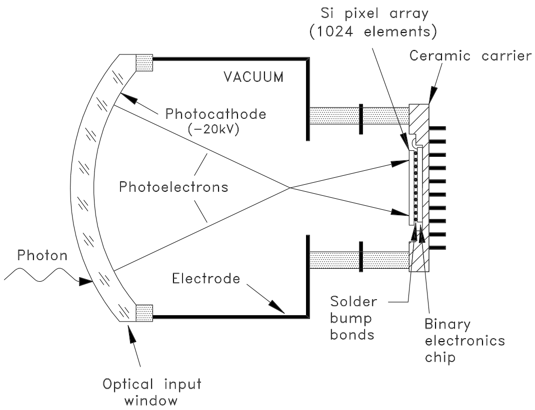

Figure 1: Schematic drawing of an HPD tube.

By changing to the Fourier conjugate domain equation 3 becomes

$$S(w;\mu,\underline{a},\underline{b}) = \left(\sum_{n=0}^{\infty} e^{-\mu}\frac{[\mu R(w;\underline{a})]^n}{n!}\right) G(w;\underline{b}) \tag{4}$$

The sum can then be rewritten as an exponential, so that $S(w;\mu,\underline{a},\underline{b})$ becomes

$$S(w;\mu,\underline{a},\underline{b}) = e^{-\mu} e^{\mu R(w;\underline{a})} G(w;\underline{b}) \tag{5}$$

Using equation 5 The fit for μ, $\underline{a}$ and $\underline{b}$ can then be done in the Fourier conjugate domain. This then removes the need to sum and any inaccuracies due to truncating the sum.

3 HPD Response Function

The single photoelectron response function is parameterised in terms of two functions, $r_f(x)$ and $r_b(x)$. $r_f(x)$ is the response for fully absorbed photoelectrons and $r_b(x)$ is the response for backscattered photoelectrons.

$$r(x) = \alpha r_b(x) + (1-\alpha) r_f(x) \tag{6}$$

Where α is the fraction of backscattered photoelectrons. The functions used for $r_f(x)$ and $r_b(x)$ are [3]

$$r_b = \begin{cases} A_1 \frac{\sin^2(a_1\pi(1-x))}{1+a_2\exp(-28.5x^2)} & x \leq 1 \\ 0 & x > 1 \end{cases} \tag{7}$$

where a_1 and a_2 are fittable parameters and A_1 is a normalisation constant.

$$r_f = A_2 \times \begin{cases} \left(1 + \left(\frac{x-1}{\sigma_{SER}}\right)^2 / a_3\right)^{-(a_3+1)/2} & x \leq 1 \\ \exp\left(-\frac{(x-1)^2}{2\sigma_{SER}^2}\right) & x > 1 \end{cases} \tag{8}$$

where a_3 and σ_{SER} are fittable parameters and A_2 is a normalisation constant.

The form of $g(x,\underline{b})$ can either be a Gaussian or be directly measured from the pedestal response.

4 Fitting

The fitting function has either 6 (measured $g(x,\underline{b})$) or 7 (Gaussian $g(x,\underline{b})$) free parameters. The program uses the NagC libraries to perform a χ^2 minimization

$$\chi^2 = \sum_i \frac{(y_i - Ns(x_i))^2}{y_i} \tag{9}$$

where y_i is the content of the ith data bin, N is a normalisation constant and $s(x)$ is obtained from

$$s(x) = \mathcal{F}^{-1}(S(w;\mu,\alpha,a_1,a_2,a_3,a_{SER},a_{PED})) \tag{10}$$

The minimization is performed on $s(x)$ not $S(w)$.

5 Computer Code

5.1 Structure

The computer code is written in C++ and requires both the NagC [4] libraries and the root [5] histogramming package. The program was developed on Linux, RedHat 6.1, with gcc version 2.95.2 and has also been used on RedHat 7.3 with gcc version 3.2.

The code consists of five main classes: Parameter, PDF, PEPDF, Noise and BPFit as well as routines for handling input of a steering text file.

- Parameter class - holds the values, names, fit limits, fit step sizes of a parameter to be determined by the fit.
- PDF class - contains a sampled version of a p.d.f. over a range with a given number of channels. Has methods to Fourier Transform and inverse FT the p.d.f. (using the NagC library) as well as operator definitions to allow adding, subtracting etc of p.d.f. objects.
- PEPDF class - holds the p.d.f. representing the single photoelectron response function. Internally uses several PDF objects of each component of the photoelectron p.d.f.
- BPFit class - main fitting object. Holds the PEPDF object, the parameter objects. Contains methods to control/set and calculate the fit.

5.2 User Interface

The program is controlled using a text file which is read in at program start up. This file is a series commands, one per line, in the form command <mandatory parameter> {optional parameter = default value}. Valid commands are

- Clock Commands
  - clock.start
    start internal timer.
  - clock.stop
    stop internal timer and print elapsed timed to standard output.
- Comparison Commands
  - comparison.plot <filename> plot data and fit to a histogram and save image as a postscript file.
- Control Commands
  - echo <setting>
    control echoing of file commands to standard output. setting is either start or stop.
  - exit
    exit the program.
- Data Commands
  - data.dir <directory>
    set directory for future file access.
  - data.load.root <filename> <histogram>
    load data from a histogram in a root file.
  - data.load.txt <filename> {channels} {offset = 0}
    load data from a text file containing floating point numbers, one per line, optionally giving the maximum number of channels to load. The n^{th} value in the file is loaded into the n+offset channel.

- `data.plot <filename>`
  plot data to a histogram and save image as a postscript file.
- `data.save <filename>`
  save data to a text file.

- Fit Commands
  - `fit.calculate`
    calculates the p.d.f. of the fit for the current set of parameter values.
  - `fit.do`
    find the values of the fit parameters which best fit the data.
  - `fit.parameters <pedestal position> <peak separation> <mu>`
    set the initial values of the fit.
  - `fit.parameter.scales <pedestal position scale> <peak separation scale> <mu scale>`
    set the initial scaling factors for the pedestal position, peak separation and mu.
  - `fit.range <first channel> <last channel>`
    set the inclusive range of channels over which the χ^2 of the fit is calculated.
  - `fit.range.ignore <first channel> <last channel>`
    set the inclusive range of channels which are ignored when calculating the χ^2 of the fit.
  - `fit.type <type>`
    set the fit type either `SINGLE_FIT` or `MULTIPLE_FIT`.

- Minimizer Commands
  - `minimizer.type <type>`
    set the minimizer type either `NAGC_SIMPLEX_MINIMIZER` or `NAGC_BOUND_MINIMIZER`.

- Noise Commands
  - `noise.global <setting>`
    set the global noise either `ON` or `OFF`. Default is `ON`.
  - `noise.global.load.txt <filename>`
    load the global noise p.d.f. from file.
  - `noise.global.parameters <sigma>`
    calculate the global noise p.d.f. using a Gaussian distribution with an initial width of `sigma` photoelectrons.
  - `noise.global.parameter.scales <sigma scale>`
    set the initial scaling factors for the global noise width. Ignored when using a global noise p.d.f. from a file.
  - `noise.pe <setting>`
    set the noise for individual photoelectrons either `ON` or `OFF`. Default is `ON`.
  - `noise.pe.load.txt <filename>`
    load the individual photoelectron noise p.d.f. from file.
  - `noise.pe.parameters <sigma>`
    calculate the photoelectron noise p.d.f. using a Gaussian distribution with an initial width of `sigma` photoelectrons.
  - `noise.pe.parameter.scales <sigma scale>`
    set the initial scaling factors for the photoelectron noise width. Ignored when using a photoelectron noise p.d.f. from a file.
  - `noise.pedestal <setting>`
    set the pedestal noise either `ON` or `OFF`. If `OFF` then the individual photoelectron noise if used. Default is `OFF`.

- noise.pedestal.load.txt <filename>
  load the pedestal noise p.d.f. from file.
- noise.pedestal.parameters <sigma>
  calculate the pedestal noise p.d.f. using a Gaussian distribution with an initial width of sigma photoelectrons.
- noise.global.parameter.scales <sigma scale>
  set the initial scaling factors for the pedestal noise width. Ignored when using a pedestal noise p.d.f. from a file.

* p.d.f. Commands
  - pdf.parameters <channels> <minimum> <maximum>
    set the properties the internal representation of the p.d.f. functions. minimum and maximum are in photoelectron units. Defaults are channels = 20000, minimum = 0 and maximum = 50.

* Result Commands
  - result.plot
    plot the results of the fit to postscript files in the current directory. data.ps is the input data, comparison.ps is the data with the fit superimposed, onepe.ps is the fit single photoelectron p.d.f., peNoise.ps is the fit individual photoelectron noise and globalNoise is the fit global noise.
  - result.print {filename}
    print the values of the fitted parameters either to the standard output or if a filename is supplied to that file.

* Scan Commands
  - scan.pedpos <parameter number> <minimum> <maximum> <step>
    set the parameter boundaries for the pedestal position.
  - scan.peaksep <parameter number> <minimum> <maximum> <step>
    set the parameter boundaries for the peak separation.
  - scan.mu <parameter number> <minimum> <maximum> <step>
    set the parameter boundaries for the mean number of photoelectrons.
  - scan.globalnoise <parameter number> <minimum> <maximum> <step>
    set the parameter boundaries for the global noise width.
  - scan.penoise <parameter number> <minimum> <maximum> <step>
    set the parameter boundaries for the photoelectron noise width.

* SER Commands
  - ser.type <type>
    sets if the fully absorbed photoelectron p.d.f. is either VARIABLE or FIXED. Default is FIXED.
  - ser.backscatter <type>
    sets if the fully absorbed photoelectron p.d.f. is either VARIABLE, FIXED or OFF. Default is FIXED.
  - ser.parameters <a_1> <a_2> <a_3> <σ> <unused> <unused>
    set the initial values of the single photoelectron p.d.f. parameters.
  - ser.parameter.scale <a_1 scale> <a_2 scale> <a_3 scale> <σ scale> <unused scale> <unused scale>
    set the initial scaling factors for the single photoelectron p.d.f. parameters.
  - ser.ratio <alpha> <unused>
    set the initial ratio between the fully absorbed photoelectrons and the backscattered photoelectrons.

– `ser.ratio.scale <alpha> <unused>`
set the initial scaling factor of the ratio between the fully absorbed photoelectrons and the backscattered photoelectrons.

5.3 Installation

The program is available as a compressed `tar` archive [6]. To compile the program first ensure that both NagC and root are both installed. Then edit the Makefile in the program source code directory changing the lines

```
ROOTSYS=/afs/cern.ch/sw/root/v3.03.09/rh72_gcc32/root
NAGC=/afs/cern.ch/sw/lhcxx/specific/rh72_gcc32/Nag_C/6.0
```

to give the location of root and NagC on the local system; then type `make`.

6 Conclusion

The 'light spectra sum rule' has been implemented in a flexible C++ program. The program has been used for fitting data from the LHCb HPDs.

References

[1] T. Gys, The pixel hybrid photon detectors for the LHCb-rich project, Nucl. Instr. and Meth. A 465 (2001) 240

[2] RICH TDR, CERN-LHCC-2000-037

[3] T. Tabarelli de Fatis, Light spectra sum rule, Nucl. Instr. and Metho. A 385 (1997) 366

[4] The NAG C Library Manual, The Numerical Algorithms Group Ltd. (2002)

[5] http://root.cern.ch/

[6] http://ppewww.ph.gla.ac.uk/~pickford/lhcb/backpulsefit-0.0.8.tar.gz