EUROPEAN ORGANIZATION FOR NUCLEAR RESEARCH

L3 Note 2157
September 11, 1997

BGO Dead Crystal Correction and Shower Fitting

I.Kominis and I.Nemenman
Princeton

Abstract

Fortran routines are developed that allow to correct for missing crystals in BGO bumps using the shower shape data from *X3* test beam at 2 and 50 GeV. Two algorithms are realized, different in both the speed of execution and the produced correction. Also, one of the algorithms (weighted χ^2 shower fitting) may be used to get better energy determination of the bumps with all crystals functioning.

1 Introduction

Many analyses in L3 are based on measurements made by the BGO calorimeter. Typically, an electromagnetic particle hititing the calorimeter starts a shower that leaves an appreciable amount of the initial particle's energy only in the crystal being hit and eight crystals surrounding it. Then the energy of the initial particle is determined by simple summation of energies deposited in these nine crystals, and by correcting this energy for the leakage of the shower between and outside the nine crystals. Also, knowing the energies deposited in these crystals, one can calculate the initial hit point within the central crystal.

The problems appear, however, when one or more of these nine crystals are not functioning, and thus the total energy, as well as the hit point, cannot be correctly calculated. This present note explains the algorithms used by the authors to deal with this problems, gives some details of the written Fortran routines, and shows preliminary results obtained with this year's 183 GeV and Z-calibration data, and 1996 Monte Carlo data. The general idea of both algoritms (to be discussed later) is as follows: a good events is described by nine numbers—energies deposited in the crystals around the hit point. These numbers are used to extract just three quantities: total deposited energy, X, and Y (coordinates of the hit within the central crystal). So, even if not all of the nine numbers are available (as is the case when one or more of the crystals is dead), we may be able to extract the three required quantities, provided that we know the shape of the shower produced by the initial particle well enough.

2 Converging Iteration Algorithm (*CIA*)

This routine corrects the energy and coordinates (θ,ϕ) of an electromagnetic bump, in case one of the crystals in the 3×3 matrix of the bump is dead. The routine has an in-built check for dead crystals, and it passes out the existing values of bump energy, θ and ϕ, when all 9 crystals are functioning.

The principle of the algorithm is as follows: two **ECL3** routines (**ECCOFG**,**ECBMCH2**) are patched together to form a routine that calculates the impact point $X = (x_1, x_2)$ in the crystal frame, when the 9 crystal energies are passed to it. So, $X = f(E_1, E_2, \ldots, E_9)$. In the case there is a dead crystal, say the third one, the initial impact point of the iteration is $X_{in} = f(E_1, E_2, E_4, \ldots, E_9)$, and obviously it is a wrong one. With this impact point, the routine **ECINTER** is called, which gives the 9 energies that should be deposited in the 3×3 matrix if the shape of the shower is electromagnetic. This gives the initial estimation of E_3, which, when put back to f, produces a new impact point X_2, closer to the real one than X_{in}. After a few such steps, the unknown energy E_3 and the impact point converge to values very close to the real ones (comparison made in Monte Carlo data). Also, in every step χ^2 of the bump is calculated. There are some cases when the routine fails to converge and the χ^2 starts growing. In these cases, the iteration stops, and the energy estimate is the one corresponding to the minimum χ^2.

From comparison with 1996 Monte Carlo data (Table. 1), one can see that the routine works pretty well, even in the case that the central crystal is missing (in this case the error in the bump energy estimation is maximum, about 10 GeV).

BUMPCORR (LBEBMP,LBECPP,S9C,ERROR,THEB,PHIB,CHI2,IFLAG,ICRYDEAD)

This is the main routine. In it there is a call to **GET_X**(CR_ENE,X).

LBEBMP,LBECPP - *INTEGER* Addresses of the bump in the **EBMP** bank.

S9C - *REAL* The corrected sum of 9 crystal energies that is passed out.

ERROR - *REAL* Estimated error in S9C

THEB,PHIB - *REAL* Corrected θ and ϕ of the bump.

CHI2 - *REAL* The value of χ^2 for the deposited energies.

IFLAG - *INTEGER* Is equal to 0 if everything went OK.

ICRYDEAD - *INTEGER* Number of missing crystal (1...9, or 0 if no crystal is dead).

GET_X (CR_ENE,X)

The routine calculates the impact point within the cetral crystal using the knowledge of deposited energies.

CR_ENE - *REAL(9)* The array of deposited energies.

X - *REAL(2)* The array of the hit coordinates.

3 buMp fitting tO Shower ShApe test Data (*MOSSAD*)

As in any constrained fit algorithm, the idea behind this one is simple. If a BGO bump is decided to be electromagnetic in nature, one may want to find that hit energy and those coordinates of the hit point which have the highest probability to produce the given depositions of energy in the crystals. These parameters will, in general, describe the energy of the bump better than a simple summation of energies in the crystals. Technically this is easily realized: the *X3* test beam data produces very reliable estimations of the shower shapes, as well as the erros on those shapes. And than one just needs to minimize χ^2 of the actual deposited energies calculated against the shower defined by three parameters: total energy and coordinates of the hit. This minimization may be done by using the **MINUIT** package from *CERN* programming library. This technique may be used to correct for missing crystals: one will fit not to all nine real energies, but eight, or seven, or even less (not less then 4), depending on how many crystals in the bump are missing. However, the main advantage of the technique is that it can also be used to correct the bumps where all crystals are operational by finding the most probable hit parameters to produce the deposited distribution of energies. Another advantage of the algorithm is that, as a result of minization, it outputs a sufficiently reliable error estimation of both the hit energy, and the coordinates of the impact point.

Of 5 routines written to realize this algorithm (**BUMP_FIT**, **NEMENROUT**, **EN9CRYINT**, **FCNCORR**, **CHECK**), only the first two should be directly invoked by the users. The other three are called internally during the minimization process. One might, however, want to change them if the use of a different shower shape database is desired, or other minimization strategy and minimization validity criteria are thought to be useful. A brief description of the routines follows:

BUMP_FIT (LBUMP,LECPP,S9C,THEB,PHIB,ERROR,CHI2,IP)

This routine is an interface routine between the user program and the main minimization routine **NEMENROUT**. It gets the necessary data from databases, calculates initial values of the parameters, and passes all the information to the actual minimization routine.

LBUMP, LECPP, S9C, THEB, PHIB, ERRO, CHI2 - The parameters are completely analogous to the parameters of **BUMP_CORR**.

IP - *INTEGER* Flag indicating the quality of the performed fit. $IP = 0$ or $IP = -1$ indicates some major problems in minimization (cf. IP in **NEMENROUT**).

NEMENROUT (E,IWE,CHI2,X,ETOT,DX,DETOT,IP)

This routine drives **MINUIT** to produce the minimization. The minimization algorthm is as follows: the initial values of parameters are defined from the input to the routine (see below). If there are no dead crystals, then minimization with respect to coordinates is perfomed, followed by minimizations with respect to energy and then to all parameters. If there are dead crystals, the energy is corrected by minimization with respect to it, and then the same three minimizations as in the case of no missing crystals are performed. The most of the time in this routine is spent during the first minimization with respect to the impact coordinates, which are highly non-linear parameters. The energy-wise minimizations are very fast, since energy is kept linear by not limiting it in **MINUIT** parameter definitions. This simplifies minimization, but requires checks that energy is not driven to unphysical limits by **MINUIT** (cf. **CHECK**).

E - *REAL(9)* An array of deposited energies on the input, and the most probable shower to produce these energies on the output.

IWE - *INTEGER(9)* An array of weights (either 0 or 1), indicating if the corresponding crystal is missing (0) and should not be included in minimization, or present (1) and should be examined. If the array contains other numbers than 0 and 1, the results will be unpredictable.

CHI2 - *REAL* On output contains the χ^2 of the best fit.

X - *REAL(2)* Should be inputed as an initial guess for the hit coordinates (cf. **GET_X**). This speeds up the fit, and produces more reliable results. On the output the variable contains the values of the coordinates producing the best shower fit.

ETOT - *REAL* On the output contains the value of the energy of the hit, producing the best fit.

DX, DETOT - *REAL* On the output contain **MINUIT** errors on the corresponding parameters.

IP - *INTEGER* On the input contains the **MINUIT** printout level desired (-1 the lowest, 3 the highest). On the output contains the estimation of the fit quality. Output value of IP equal to $0 \ldots 4$ corresponds to the same values of the ISTAT variable of **MNSTAT MINUIT** call. Values above 0 say that the fit was succesfull, while 0 indicates that no minimization has been done. In the worst cases, when the minimization drifts the values of parameters to unphysical or boundary regions, IP is set to -1, and the initial values are returned at the output.

EN9CRYINT (X,ETOT,E,DE,ISTAT)

This is an interface routine, which extracts the energy deposition distributions from the banks. It's called internally by both **FCNCORR** and **NEMENROUT**. The purpose of this routine is to make it easier to change the shower databases, error estimations calculations, direction of the axis, etc. if necessary. To make all of these changes, one will just need to change a few lines in **EN9CRYINT**, not touching any other routines.

X - *REAL(2)* Input coordinates of the generated shower.

ETOT - *REAL* Energy of the hit that starts the shower.

E - *REAL(9)* Estimated shower shape.

DE - *REAL(9)* Estimated errors of the shower shape.

ISTAT - *INTEGER* If it is nonzero, the database is not present.

FCNCORR (NPAR,DVGRAD,DCHI2,DVPAR,IFLAG)

This is actually the χ^2 calculation routine which is minimized by **MINUIT**. The description of parameters of the routine can be found in **MINUIT** manual.

***LOGICAL* CHECK** (ETOT)

The routine checks whether the minimization is proceeding smoothly. It is called after every minimization step. It returns .TRUE. if the minimization is still within the physical limits, and .FALSE. otherwise. The checks, made in the routine are:

- hit point is far (0.1 mm away) from the borders of the crystals;
- during the minimization the energy has decreased by not more than a factor of 2;
- energy has not been increased by more than a factor of 20 (it cannot increase more even if the most energetic crystal is missing).

Usually, if the routine returns .FALSE., it's due to the first check.

ETOT - *REAL* The actual measured energy in all nine crystals.

4 Comparison of Two Algorithms

Though both rely on good knowledge of shower shapes, the algorithms are drasticly different. The *CIA* algorithm usually converges in three or four steps, thus it accesses shower databases very few times. On the contrary, *MOSSAD* algorithm, accesses the shower databases, at least, two hundred times. In addition to that, a lot of time is lost in **MINUIT** while calculating derivatives of the function and making decisions about future progress of minimization. The authors estimate that *MOSSAD* algorithm is, on average, a hundred to two hundred times slower than *CIA* algorithm. This estimation, however, was not explicitely verified.

The *CIA* algorithm has one more advantage over *MOSSAD*. If the impact at the detector occured close to the crystal boundaries, then minimization might take much more time than usually, and might even not converge. This never happens in *CIA* algorithm, thus making it more reliable.

The complexity of *MOSSAD*, which is the source of its major problems, also creates advantages. As was already said, the algorithm may be used to correct patterns with no dead crystals, or with more than one dead crystal. And, if it manages to correct for missing crystals successfully, its output is usually better than the one of *CIA*.

Thus the authors of the programs believe that both algorithms must be used complementary. If speed is crucial, *CIA* should be used on events with missing crystals. If energy resolution is more important, *MOSSAD* should be used on all of the events. And when *MOSSAD* fails to do the job ($IP = -1$), *CIA* should take over and correct for dead crystals.

5 183 GeV, Z-Calibration, And Monte Carlo Results

The algorithms have been tested on 183 GeV and Z-calibration data from 1997 run, and on 1996 Monte Carlo.

In the real data, the EE string selection of Bhabha events was applied, as well as the selection cuts from [1]. This gave 61 high energy (183 GeV) Bhabha events and 2380 Z-calibration events.

In the Fig.1 one may find a Bhabha peak extracted by the mentioned cuts from 183 GeV data. The number of Bhabha events are plotted versus $(E_{bump1} + E_{bump2})/\sqrt{(s)}$. There are 61 events passing the cuts in the barrell, and the characteristic width of the Bhabha peak is 1.8%.

In the Fig.2 the same distribution has been corected by the *CIA* algorithm. We expect 1% of all the crystals in BGO to be dead, thus 8of bumps will have one of side crystals missing. 61 events passing the cuts form 122 bumps, out of which 10 contain missing crystals. This agrees well with the expectations. Out of these 10 bumps four are moved *significantly* by **BUMPCORR**. This produces slight decrease in the width of Bhabha peak, and shifts it towards the expected central value of 1.0.

Finally, Fig.3 shows *MOSSAD's* correction of the distribution. It's moved still closer to the central value of 1.0, and the resolution is 1.35% —a lot better that the uncorreceted one.

So, the 183 GeV data appears to show the use of both algorithms. But as Fig.4 shows, at Z-calibration energy the **BUMPCORR** correction is just slightly better than the uncorrected data, and **BUMP_FIT** output is the worst. The reason for uncorrected data to be the best at this energy is unknown, but the best guess is that the calibration of BGO was done at 45 GeV, correcting for missing crystals by recalibration of their neighbors. This will mean a deviation from the expected electromagnetic shower shape, and thus explain the relatively poor performance of both algorithms, which heavily depend on its correct knowledge. But, combining with the 183 Gev results, this means that **BUMP_FIT** may be used for fixing calibration errors.

The Monte Carlo run with perfect calibration and one crystal dead in every most energetic Bhabha bump again shows the consequetive improvement from uncorrected resolution to *CIA* correction and, finally, *MOSSAD* correction. The Table. 1 shows the parameters of the Bhabha peak in all of the cases, as determined by fitting it to the Gaussian distribution.

	Height of Peak	Center of Peak	Width of Peak
Monte Carlo, all Crystals Functioning	363.5	1.004	0.82%
Monte Carlo, One Crystal Dead	150.1	0.9878	1.31%
BUMPCORR Restoration	271.1	1.005	0.92%
BUMP_FIT Restoration	284.0	1.006	0.83%

Table 1: Monte Carlo Bhabha Resolution Parameters

Summarizing the above said, we may conclude that the amount of data processed does not allow to make any final conclusions about the performance of the algorithms. So more data has to be analysed, and the routines have to be further developed.

References

[1] D. Bourilkov. *L3 note 1301.*

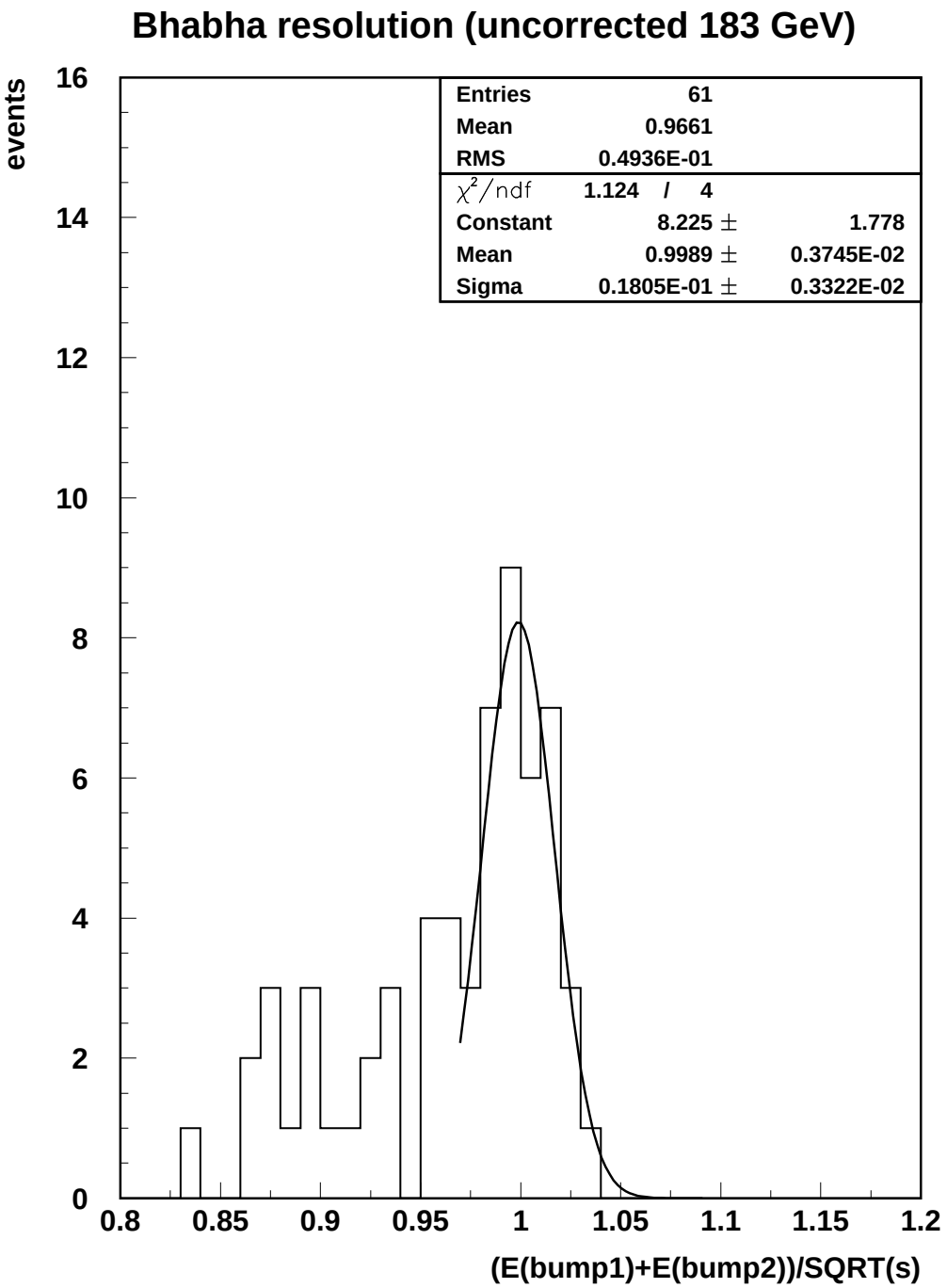

Figure 1: 183 GeV Bhabha Resolution, Uncorrected.

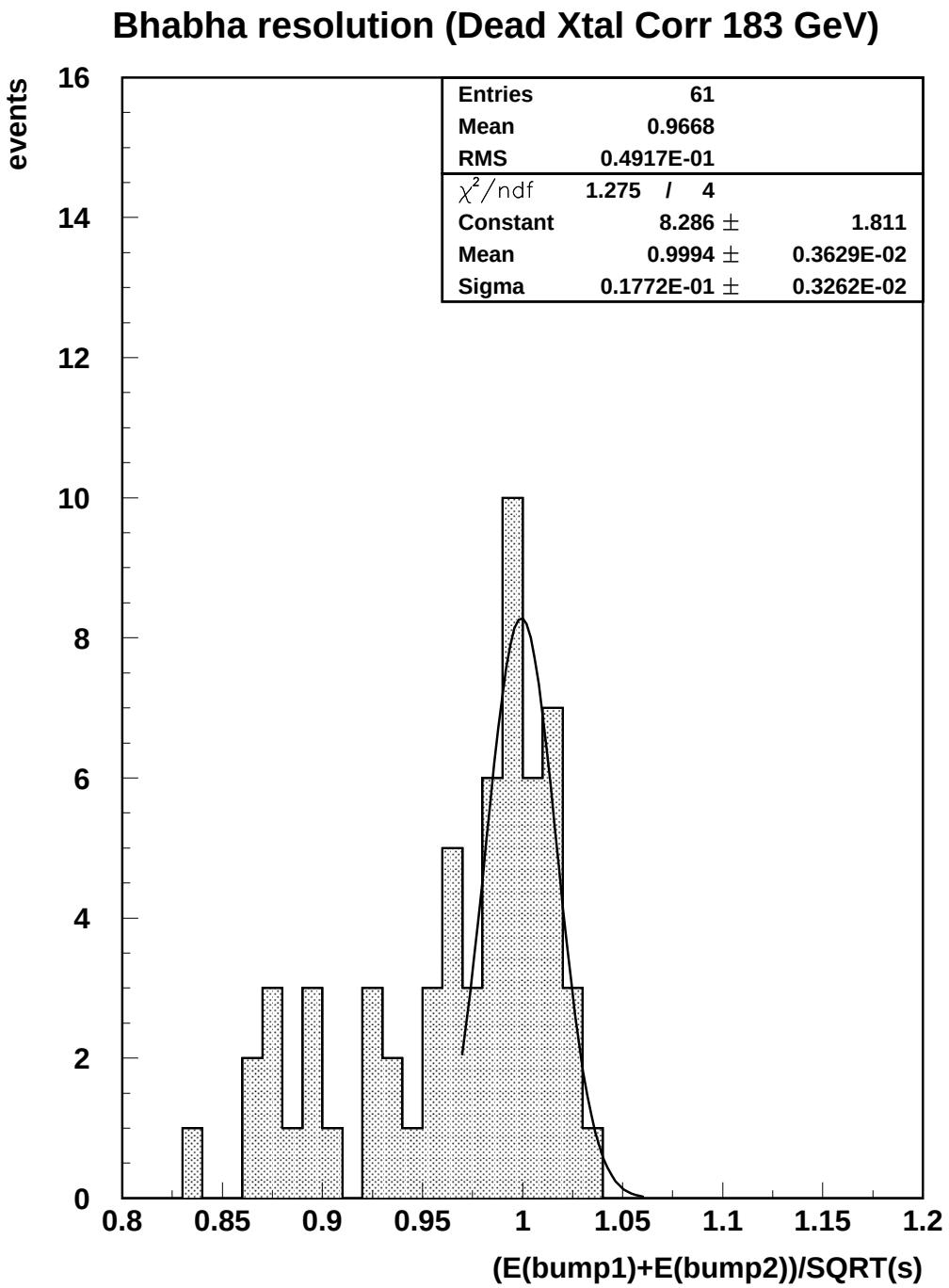

Figure 2: 183 GeV Bhabha Resolution, CIA corrected.

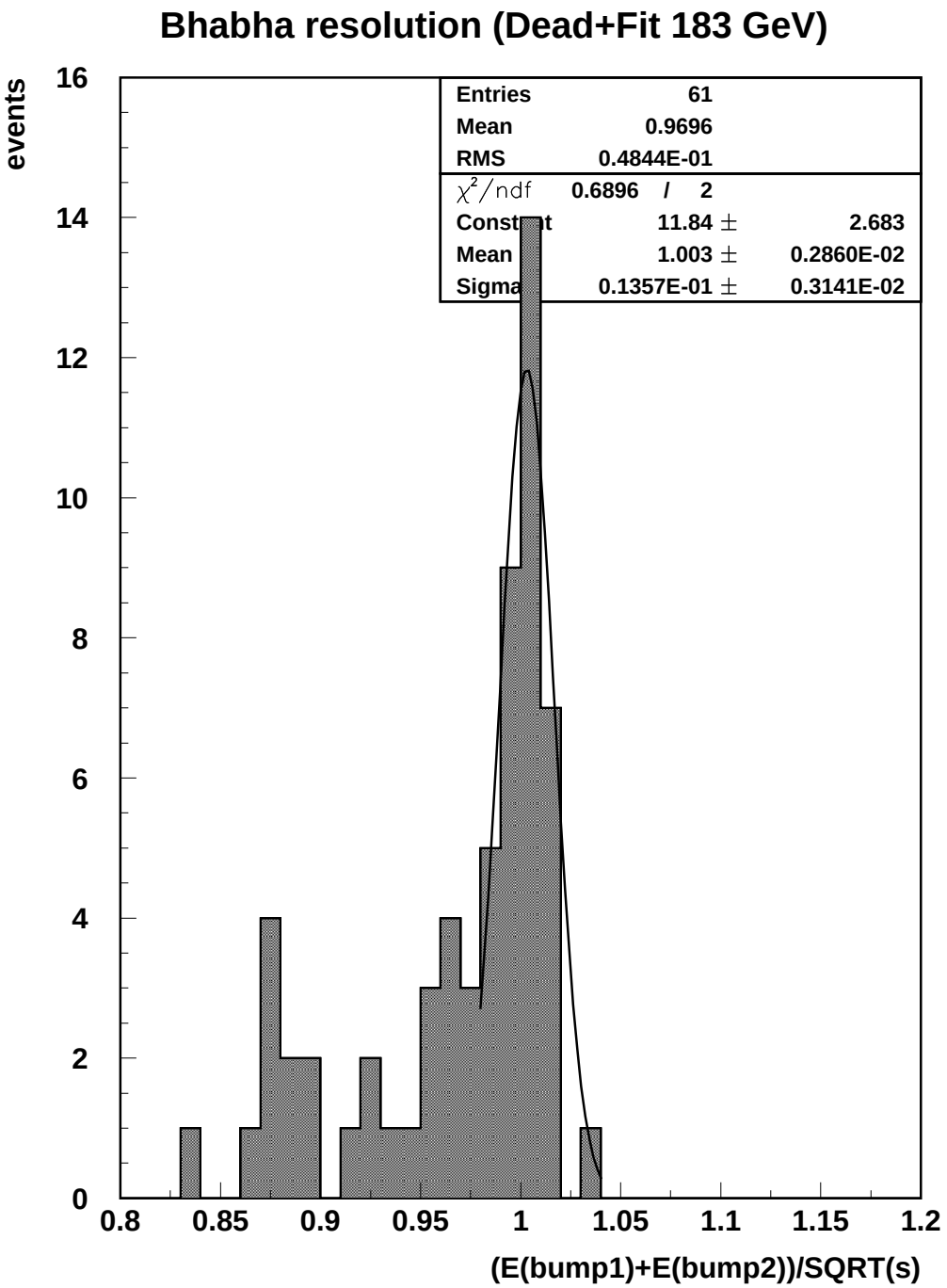

Figure 3: 183 GeV Bhabha Resolution, MOSSAD corrected.

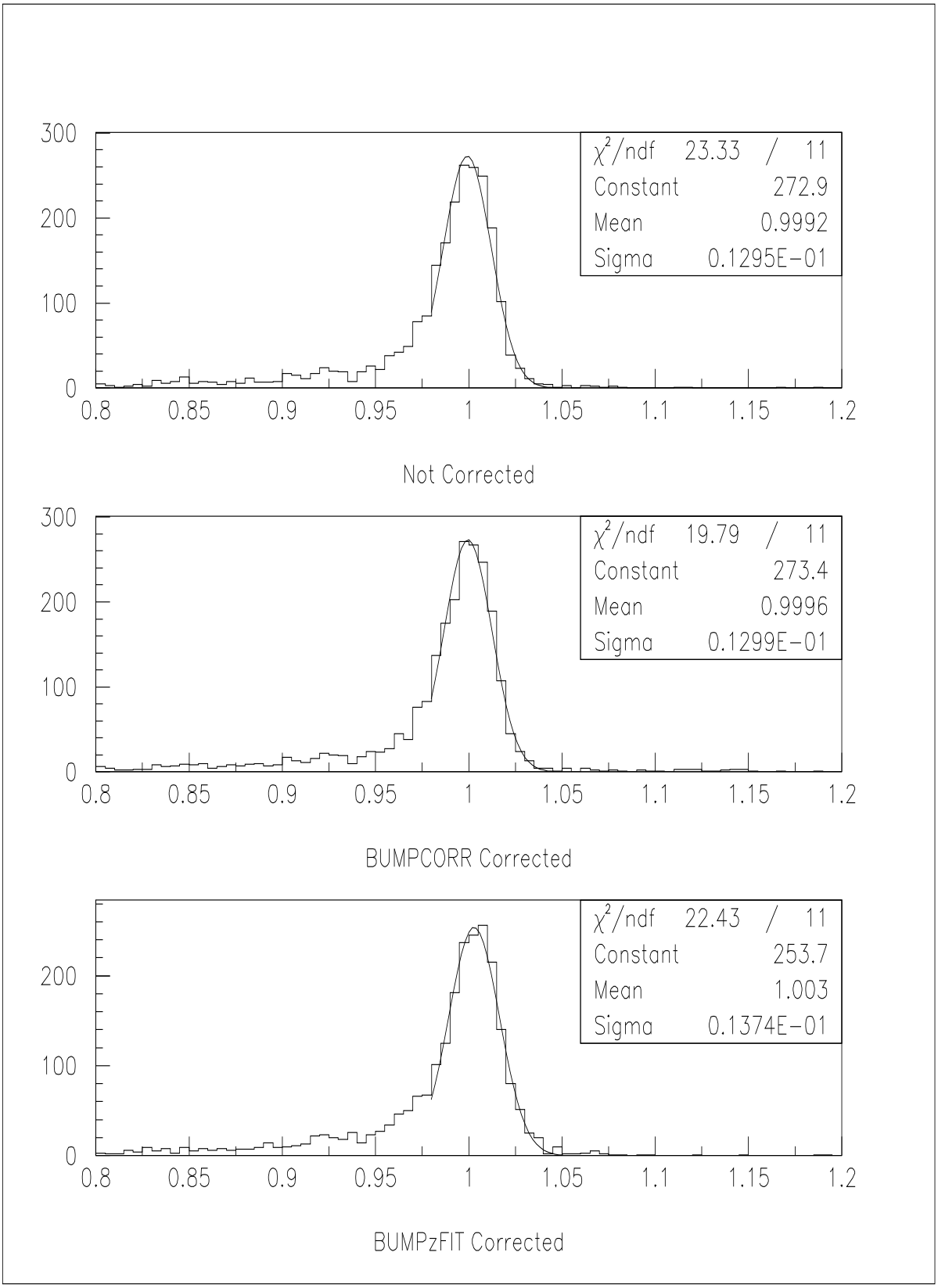

Figure 4: Z-Calibration Bhabha Resolution.