The Elastic Fitter
SNO-STR-95-040

S.J. Brice, Oxford University

August 30, 1995

Abstract

This paper introduces a new type of vertex fitter for SNO. It is a development and extension of the Elastic Arms algorithm of Ohlsson, Peterson, and Yuille [Ohlsson 92, Ohlsson 93] and can be regarded as a fancy time fitter that uses a simulated annealing technique in the minimisation and has a very appealing method for eliminating noise and reflection hits as the fit proceeds. It is shown that the elastic fitter outperforms both the time and quad fitters but the improved quality of fit is at the expense of a more complicated algorithm with a number of free parameters which need to be set. It may well prove to be the case that the Elastic Fitter does not significantly improve upon existing methods, but the techniques used by the fitter are certainly of great interest and warrant further study.

1 Introduction

The motivation for investigating a new kind of vertex fitting has not been so much to improve the quality of a fiducial volume cut or eliminate PMT $\beta\gamma$ events, but rather to produce the best possible fit position and time that can then be used as an origin for extracting hit pattern parameters for event classification. A small change in fit position can produce fairly significant changes in the parameters that are fed into a neural network (see [Brice 95] for a description of the use of a neural network in SNO). In fulfilling this assigned role the Elastic Fitter has proven pretty successful. Its performance and ease of use when applied to the more standard fitter tasks may not show such success but are worth looking at.

2 The Algorithm

The Elastic Fitter algorithm is a development of the Elastic Arms algorithm of Ohlsson, Peterson, and Yuille [Ohlsson 92, Ohlsson 93] and these references should be consulted for a full introduction to the technique. The Elastic Arms algorithm was designed to fit TPC data to straight or helical tracks. It has been changed here to perform a combined position and direction fit with the respective minimisations being largely decoupled. The next section will describe the algorithm in general terms with the following two sections dealing with its application to a position and to a direction fit respectively.

2.1 A General Description

Given a set of hits (TPC, PMT, etc.) labelled by i and forming an event, the task is to fit them to some track/vertex defined by a set of parameters π_k. To do this one first constructs the square of the minimum distance from each hit to the track/vertex. This minimum distance is assumed to be Gaussian distributed about zero and its square is denoted by M_i. The desired fit could then most simply be carried out by minimising

$$\chi^2 = \sum_i \frac{M_i}{\sigma_i^2} \tag{1}$$

with respect to the parameters π_k, where σ_i is a measure of the Gaussian error on each hit. Such minimisations are plagued by problems with local minima and, more seriously, by the effects of noise and 'bad' hits. But suppose, through divine insight, it were possible to identify these bad hits. One could then define a weight S_i for each hit where

$$\begin{aligned} S_i &= 1 \quad \text{for a true hit} \\ S_i &= 0 \quad \text{for a bad hit} \end{aligned}$$

and then minimise the quantity

$$E' = \sum_i S_i M_i \tag{2}$$

where the summation still extends over all the hits in an event and the σ_i are all assumed to be the same.

In the absence of divine insight, suppose some method can be constructed to enable the algorithm to decide for itself which hits are good and which are bad and set S_i accordingly. In this case the function of Equation 2 is the wrong function to minimise, as its global minimum can most easily be achieved by saying that EVERY hit is bad and setting $S_i = 0$ for then all. What is needed is a penalty term in the function which ensures that there is some cost to assigning a bad hit. Calling this cost λ the correct minimisation function is

$$E = \sum_i [\, S_i M_i + (1 - S_i)\lambda \,] \tag{3}$$

The optimal fit parameters π_k for a given event are those which minimise E and the simplest way to do that is iteratively using gradient descent. Starting with some initial values of π_k the fit parameters are successively altered by $\Delta\pi_k$ given by

$$\Delta\pi_k \equiv -\eta \frac{\partial E}{\partial \pi_k} = -\eta \sum_i S_i \frac{\partial M_i}{\partial \pi_k} \tag{4}$$

where η is an update parameter controlling the speed of descent.

It is now necessary to construct a method by which the algorithm can set the weights S_i for itself. The first step is to notice the close analogy between the minimisation function of Equation 3 and the total energy of a system consisting of N particles each of which has two energy states available to it (a physical picture would be that of a lattice of spin $\frac{1}{2}$ particles). In this case $S_i = 1$ would indicate the occupation by the i^{th} particle of the level with enegy M_i and $S_i = 0$ the occupation of the level with energy λ. The λ level has fixed energy, but the M_i level has an energy that depends upon the occupations and energies of all the other particles and is also different for each particle. Pushing the analogy further one can consider the system of particles existing in an environment of finite temperature. To calculate various properties of the system of particles it is usually only necessary to consider the average energy

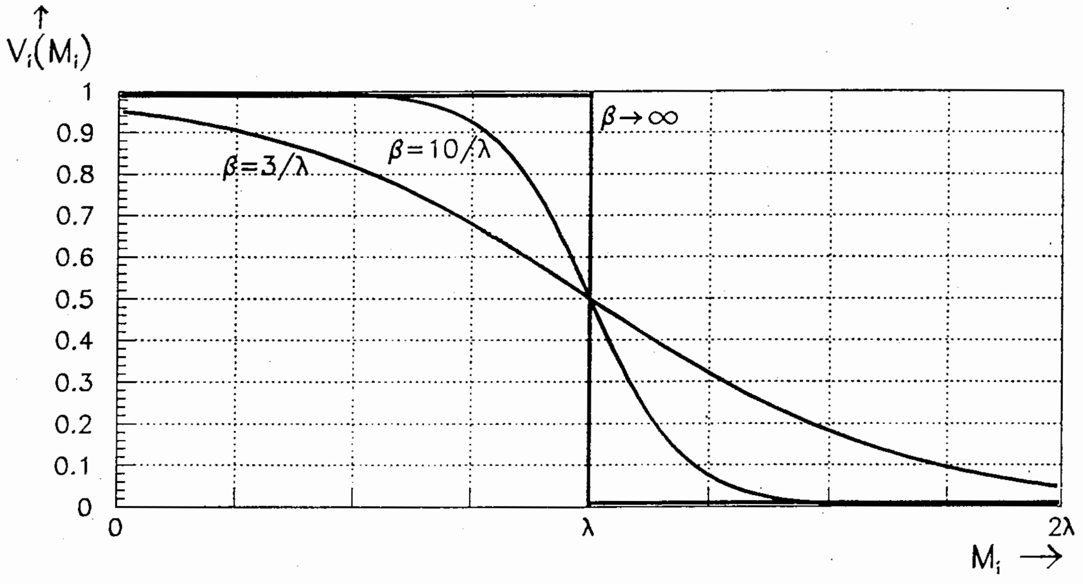


Figure 1: The weighting factor V_i as a function of the squared distance measure M_i for various values of the temperature parameter β

level occupation $< S_i >_T$ at a particular temperature and this is done by using the Boltzmann distribution. With $\beta \equiv \frac{1}{T}$, one can therefore define

$$\begin{aligned} V_i \equiv < S_i >_T &= \frac{1 \times e^{-\beta M_i} + 0 \times e^{-\beta\lambda}}{e^{-\beta M_i} + e^{-\beta\lambda}} \\ &= [1 + e^{-\beta(\lambda - M_i)}]^{-1} \end{aligned} \tag{5}$$

One can now take the results of pushing the energy level analogy back into the the fitting algorithm and instead of using S_i in the gradient descent use $< S_i >_T$. So that

$$\Delta\pi_k = -\eta \sum_i V_i \frac{\partial M_i}{\partial \pi_k} \tag{6}$$

where

$$V_i = [1 + e^{-\beta(\lambda - M_i)}]^{-1} \tag{7}$$

This weighting factor V_i can be derived in many ways (the working above is but one of them) and is the key to the algorithm. It is shown in Figure 1 as a function of M_i for various values of the temperature parameter β. Values of M_i less than λ receive a weight close to 1, whereas values above λ are effectively discounted from the fit with a weight close to 0. It can be seen that λ is behaving as a threshold, with hits whose M_i is greater than this threshold being discounted as bad hits. Notice also that the sharpness of the threshold increases as the temperature T is lowered (i.e. as β is raised). This suggests an amendment to the minimisation strategy. At the start of the gradient descent iteration the initial values of the fit parameters π_k may not be very accurate and so a slow threshold between good and bad hits (i.e. a high temperture T_{initial}) is necessary so that true hits are not completely rejected merely because of this inaccuracy. As iteration proceeds and the fit improves then this *elasticity* in the threshold (the property which gives the fitter its name) should be reduced (i.e. T should be lowered) so that hits are neglected whose M_i we can now be confident are bad. This lowering of temperature as the fit proceeds should not continue to $T = 0$ but stop at some final temperature T_{final} that reflects our uncertainty in the value of λ that should be used. This proceedure is an example of simulated annealing (the name comes from the analogy with the slow cooling of a metal to give it greater strength). It is becoming a widely used method principally because it greatly increases the chance of finding the global rather than a local minimum of a function.

There is one further detail needed to complete the algorithm. Close to a minimum the minimisation function E has an approximately parabolic shape, In order to avoid oscillation from one side of the minimum to the other, as the iteration nears its end, it is necessary to progressively lower the step length parameter η. This should only start once T_{final} has been reached and will enable faster convergence.

2.2 The Position Fit

To apply the Elastic algorithm to a specific situation one need only specify the squared distance measure M_i and the fit parameters to be obtained. For the position fit M_i^{pos} is simply the squared residual for each hit, where a constant δ^{pos} is subtracted from the residual to try to ensure that the residuals for the true vertex are distributed about zero.

$$M_i^{\text{pos}} = [\ |\underline{\mathbf{r}}_i - \underline{\mathbf{r}}_{\text{fit}}| - v(t_i - t_{\text{fit}}) - \delta^{\text{pos}}\ ]^2 \tag{8}$$

where $\underline{\mathbf{r}}_i, t_i$ are the PMT position and time, $\underline{\mathbf{r}}_{\text{fit}}, t_{\text{fit}}$ are the fit position and time (the fit parameters one wishes to obtain), and v is the speed of light in H_2O.

As an aside the χ^2 that the time fitter minimises is simply

$$\chi^2 = \frac{1}{\sigma^2}\sum_i M_i^{\text{pos}}$$

where σ is v times the timing jitter of the PMTs.

Plugging the definition of M_i^{pos} into equation 6 the position fit proceeds by first picking some initial values for $\underline{\mathbf{r}}_{\text{fit}}, t_{\text{fit}}$ and then updating them with each iteration according to

$$\begin{aligned} \Delta\underline{\mathbf{r}}_{\text{fit}} &= +2\eta^{\text{pos}}\textstyle\sum_i V_i^{\text{pos}}\ [\ |\underline{\mathbf{r}}_i - \underline{\mathbf{r}}_{\text{fit}}| - v(t_i - t_{\text{fit}}) - \delta^{\text{pos}}\ ]\ \dfrac{\underline{\mathbf{r}}_i - \underline{\mathbf{r}}_{\text{fit}}}{|\underline{\mathbf{r}}_i - \underline{\mathbf{r}}_{\text{fit}}|} \\ v\Delta t_{\text{fit}} &= -2\eta^{\text{pos}}\textstyle\sum_i V_i^{\text{pos}}\ [\ |\underline{\mathbf{r}}_i - \underline{\mathbf{r}}_{\text{fit}}| - v(t_i - t_{\text{fit}}) - \delta^{\text{pos}}\ ] \end{aligned} \tag{9}$$

with V_i^{pos} defined as before to be

$$V_i^{\text{pos}} = \left[\ 1 + e^{-\beta^{\text{pos}}(\lambda^{\text{pos}} - M_i^{\text{pos}})}\ \right]$$

Iteration proceeds with T^{pos} (i.e. $1/\beta^{\text{pos}}$) decreasing from $T_{\text{initial}}^{\text{pos}}$ to $T_{\text{final}}^{\text{pos}}$ and then with η^{pos} being lowered.

2.3 The Direction Fit

To define M_i^{dir} for the direction fit one starts by specifying $R(\theta_{\text{fit}}, \phi_{\text{fit}})$ as the matrix that rotates a coordinate system so that its polar axis is along the Čerenkov cone axis (which is

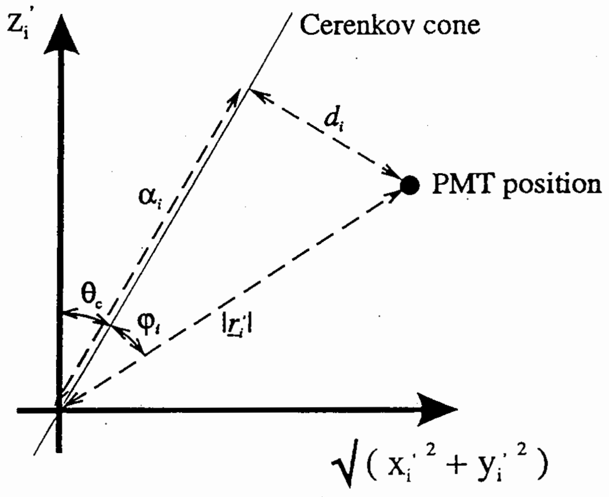


Figure 2: The transformed space where the PMT hit is at $\underline{\mathrm{r}}_i'$, the diagonal line is the Čerenkov cone, d_i is the nearest distance from the PMT hit to the cone, and ϕ_i is the smallest angle between the Čerenkov cone and the PMT hit.

given by θ_{fit} and ϕ_{fit} in detector coordinates). One can then define a transformed PMT hit position $\underline{\mathrm{r}}_i'$

$$\underline{\mathrm{r}}_i' = R(\theta_{\text{fit}}, \phi_{\text{fit}})(\underline{\mathrm{r}}_i - \underline{\mathrm{r}}_{\text{fit}}) \tag{10}$$

In this transformed coordinate system the fit position is at the coordinate origin and the Čerenkov cone axis lies along the polar axis. This is shown in Figure 2 which also indicates the minimum PMT-cone distance d_i. Some simple geometry leads to

$$d_i = \cos\theta_c\sqrt{(x_i'^2 + y_i'^2)} - \sin\theta_c z_i' \tag{11}$$

This means the sin of the angle ϕ_i between the Čerenkov cone and the PMT hit is

$$\sin\phi_i = \frac{d_i}{|\underline{\mathrm{r}}_i'|} = \frac{\cos\theta_c\sqrt{(x_i'^2 + y_i'^2)} - \sin\theta_c z_i'}{\sqrt{(x_i'^2 + y_i'^2 + z_i'^2)}} \tag{12}$$

Also show in Figure 2 is the distance α_i along the cone of the cone point nearest to the PMT hit position. It is given by

$$\alpha_i = \sin\theta_c\sqrt{(x_i'^2 + y_i'^2)} + \cos\theta_c z_i' \tag{13}$$

PMT hits which have a negative alpha, i.e. hits which are behind the cone should be left out of the fit as they can badly skew the result. One can now define the minimum squared distance M_i^{dir} as $\sin^2\phi_i$ i.e.

$$\begin{aligned} M_i^{\text{dir}} &= \frac{[\,\cos\theta_c\sqrt{(x_i'^2 + y_i'^2)} - \sin\theta_c z_i'\,]^2}{[\,x_i'^2 + y_i'^2 + z_i'^2\,]} \qquad && \alpha > 0 \\ &= \infty && \alpha < 0 \end{aligned} \tag{14}$$

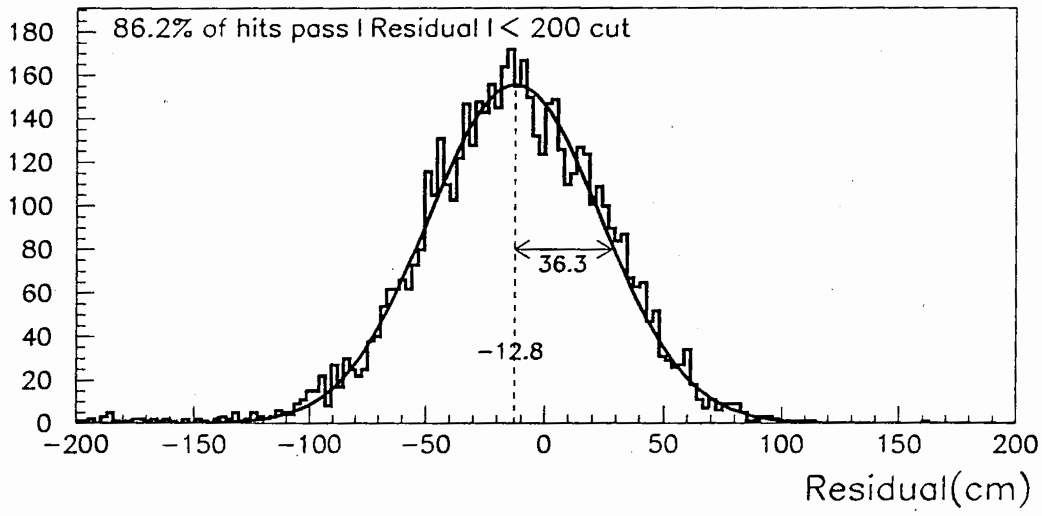


Figure 3: The distribution of the residual $|\underline{\mathbf{r}}_i - \underline{\mathbf{r}}_{\text{fit}}| - v(t_i - t_{\text{fit}})$ for the true vertices of 100 charged current events isotropic in the D_2O.

With M_i^{dir} defined one can once again plug it into Equation 6 and produce the iterative update rules for the fit parameters θ_{fit} and ϕ_{fit}. With $\alpha_i < 0$ then $\Delta\theta_{\text{fit}} = \Delta\phi_{\text{fit}} = 0$, and with $\alpha_i > 0$ the updates are

$$\begin{aligned} \Delta\theta_{\text{fit}} &= -\eta^{\text{dir}} \textstyle\sum_i V_i^{\text{dir}} V_i^{\text{pos}} \dfrac{2d_i}{\sqrt{(x_i'^2 + y_i'^2)}} \dfrac{\underline{\mathbf{r}}_i - \underline{\mathbf{r}}_{\text{fit}}}{|\underline{\mathbf{r}}_i - \underline{\mathbf{r}}_{\text{fit}}|^2} \bullet \dfrac{\partial R^{\text{T}}}{\partial\theta_{\text{fit}}} \begin{pmatrix} x_i' \cos\theta_c \\ y_i' \cos\theta_c \\ -\sqrt{(x_i'^2 + y_i'^2)} \sin\theta_c \end{pmatrix} \\ \Delta\phi_{\text{fit}} &= -\eta^{\text{dir}} \textstyle\sum_i V_i^{\text{dir}} V_i^{\text{pos}} \dfrac{2d_i}{\sqrt{(x_i'^2 + y_i'^2)}} \dfrac{\underline{\mathbf{r}}_i - \underline{\mathbf{r}}_{\text{fit}}}{|\underline{\mathbf{r}}_i - \underline{\mathbf{r}}_{\text{fit}}|^2} \bullet \dfrac{\partial R^{\text{T}}}{\partial\phi_{\text{fit}}} \begin{pmatrix} x_i' \cos\theta_c \\ y_i' \cos\theta_c \\ -\sqrt{(x_i'^2 + y_i'^2)} \sin\theta_c \end{pmatrix} \end{aligned} \tag{15}$$

where R^{T} is the transpose of $R(\theta_{\text{fit}}, \phi_{\text{fit}})$.

One deviation from the general algorithm occurs in the equations above. The position weighting factor V_i^{pos} has been included in addition to the direction weighting factor V_i^{dir}. This makes sense as hits which the position fit declares to be bad should also be excluded from the direction fit. The direction fit has not been allowed to impact the position fit in the same way because it is significantly less precise.

3 Setting the Free Parameters

The largest single disadvantage of the Elastic fitter is the number of free parameters controlling the algorithm which need to be externally set. Some have a strict dependence on the physics and properties of the detector, others reflect the average properties of the minimisation function. These parameters are listed below with indications of how they should be set.

Position Fit

$T^{\text{pos}}_{\text{initial}}$:- This is the initial temperature of the simulated annealing. There are no strict guidelines to determine how it should be set, but a value of 10-50 times the final temperature $T^{\text{pos}}_{\text{final}}$ is a good bet.

$T^{\text{pos}}_{\text{final}}$:- The final temperature of the simulated annealing can be set with reference to Equation 5. The exponential in V^{pos}_i is effectively a Gaussian in $\pm\sqrt{M^{\text{pos}}_i}$ and so $\sqrt{T^{\text{pos}}_{\text{final}}}$ should be the width of this Gaussian. Figure 3 shows the distribution of the residual (i.e. $\pm\sqrt{M^{\text{pos}}_i}$) for the true vertex position of 100 electrons with a charged current energy spectrum. The width indicated is 36.3cm (which is just v times the timing jitter of the PMTs), meaning that $T^{\text{pos}}_{\text{final}}$ should be set to $\sim$1300cm^2.

κ^{pos}_T :- This is the rate at which the temperature T^{pos} decreases. From one iteration to the next the annealing temperature decreases by a factor of κ_T. Once again there are no strict guidelines, but a value of 0.985 works well.

λ^{pos} :- Events with squared residuals (i.e. M^{pos}_i) greater than λ^{pos} should be considered bad hits. From Figure 3 it can bee seen that a good value for this residual is 200cm, meaning that λ^{pos} should be set at 40000cm^2.

δ^{pos} :- As stated before the algorithm works best if the distance measure being minimised is Gaussian distributed about zero. As can be seen in Figure 3 the residual for a true vertex is nicely Gaussian distributed but has an offset of -12.8cm. This offset is a manifestation of the ever present fitter pull and to try to counter it δ^{pos} should be set to -12.8cm.

η^{pos} :- The size of the iteration step length parameter η^{pos} is dependent upon the shape of the minimisation function. Experimentation shows that a value of 0.01 works well.

κ^{pos}_η :- This is the rate at which the step length η^{pos} decreases once $T^{\text{pos}}_{\text{final}}$ has been reached. From one iteration to the next the η^{pos} decreases by a factor of κ^{pos}_η. A value of 0.985 is suitable.

initial $\underline{\mathbf{r}}_{\text{fit}}$, t_{fit} :- One has to provide the algorithm with starting values of the fit variables. These should be produced by another fitter. With the Elastic fitter being simply a fancy Time fitter, using an initial Time fitted vertex is unsuitable as a bad hit which pulls a Time vertex is also likely to pull an Elastic vertex if it is allowed to initially dominate. A Quad fitted vertex is suitable, however.

$ftol^{\text{pos}}$:- The iteration should be stopped once the minimisation function E differs fractionally by less than $ftol$ from one iteration to the next. A suiable value for $ftol^{\text{pos}}$ is 0.000001.

Direction Fit

$T^{\text{dir}}_{\text{initial}}$:- As with the position fit a value of 10-50 times the final temperature $T^{\text{dir}}_{\text{final}}$ is a good bet.

$T^{\text{dir}}_{\text{final}}$:- Figure 4 shows the distribution of $\sin\phi_i$ (i.e. $\pm\sqrt{M^{\text{dir}}_i}$) for the true vertex direction of 100 electrons with a charged current energy spectrum. The width indicated is 0.26, meaning that $T^{\text{dir}}_{\text{final}}$ should be set to 0.07.

κ^{dir}_T :- For the direction temperature decrease rate there are, again, no strict guidelines, but a value of 0.985 works well.

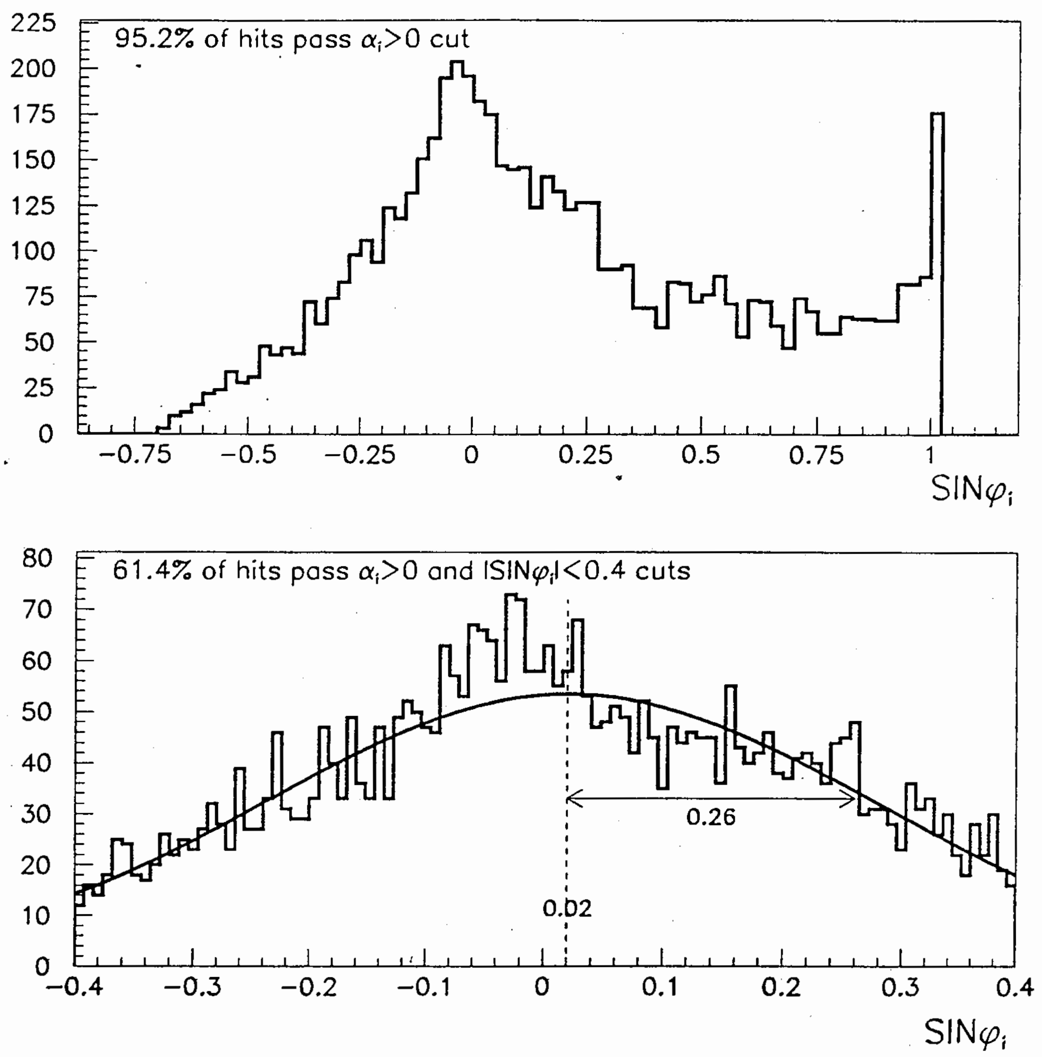


Figure 4: The distribution of SIN of the smallest angle between the PMT hit position and the Čerenkov cone for the true vertices of 100 charged current events isotropic in the D_2O.

λ^{dir} :- Events with $\sin^2\phi_i$ (i.e. M_i^{dir}) greater than λ^{dir} should be considered bad hits. From Figure 4 it can bee seen that a good value for the $\sin\phi_i$ cutoff is 0.4, meaning that λ^{dir} should be set at 0.16.

η^{dir} :- Experimentation shows that a value of 0.1 works well.

κ_η^{dir} :- From one iteration to the next the η^{dir} decreases by a factor of κ_η^{dir}. A value of 0.985 is again suitable.

initial θ_{fit}, ϕ_{fit} :- Multiple scattering ensures that the direction fit is unavoidably less accurate than the position fit. The accuracy of the initial values of θ_{fit}, and ϕ_{fit} is therefore less important than that of their position fit counterparts. A simple mean of the hit directions from some fitted vertex is sufficient.

$ftol^{\text{dir}}$:- This can be set to be the same as $ftol^{\text{pos}}$, i.e. to 0.000001.

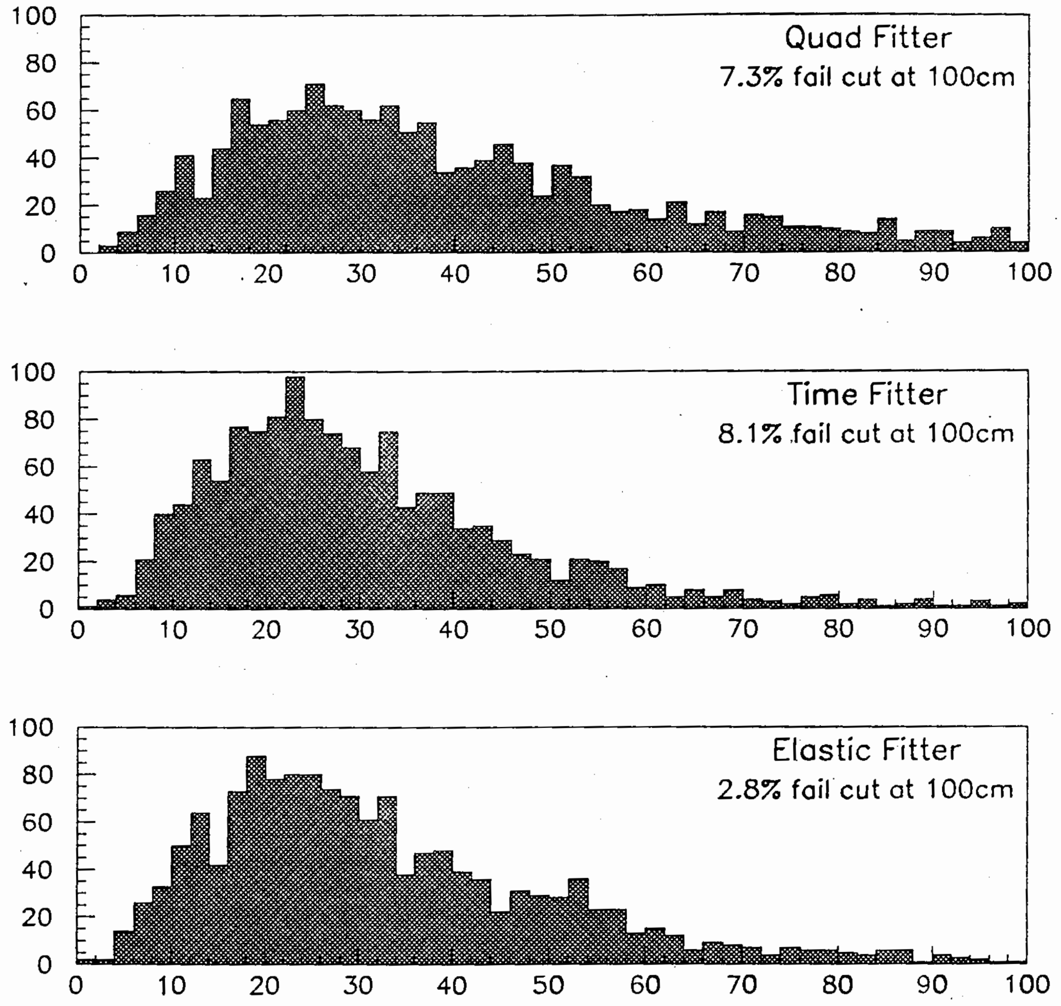


Figure 5: The distance (in cm) between the fitted and true vertices for 1500 5MeV electron events isotropic in the D2O and using the Quad Fitter, the Time Fitter with an initial Quad vertex, and the Elastic Fitter with the same initial Quad vertex. A cut is made at 100cm

4 Results

All results presented in this paper are generated using SNOMAN version 2.08. Shown in Figures 5 and 6 are the results of fitting 1500 5MeV electron events isotropically distributed in the D2O (the number of events which fail to fit is negligible). The histograms show the distance from true to fitted vertex for the Quad Fitter, the Time Fitter with an initial Quad vertex, and the Elastic Fitter with the same initial Quad vertex. Figure 5 has a cut at 100cm and Figure 6 a cut at 300cm.

When comparing fitters via these distance error histograms two criteria are relevant:-

1) The extent of the tail out to large distances

2) The proximity of the main peak to zero.

A long tail is a result of bad hits being incorporated in a fit and skewing the result whereas the location and sharpness of the main peak depends loosely upon how much of the available hit

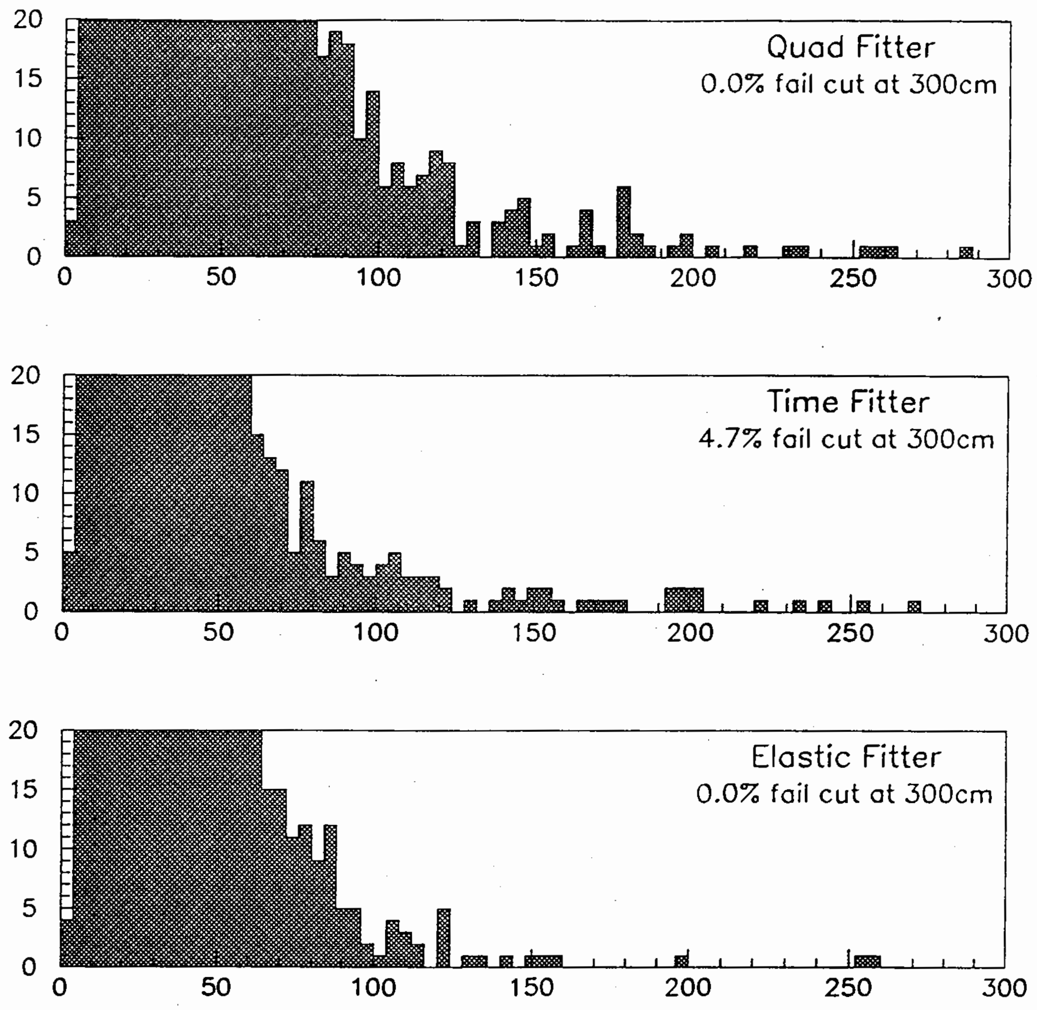
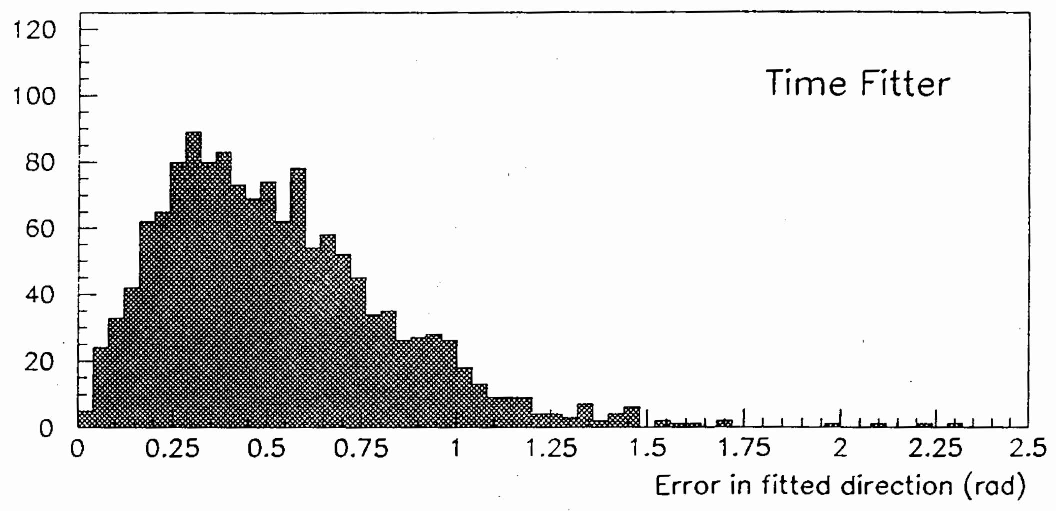


Figure 6: The distance (in cm) between the fitted and true vertices for 1500 5MeV electron events isotropic in the D2O and using the Quad Fitter, the Time Fitter with an initial Quad vertex, and the Elastic Fitter with the same initial Quad vertex. A cut is made at 300cm

information is used by the fitter (here there is a vague analogy with a $\sqrt{N}$ statistical error).

Typically the Quad Fitter [Frati 94b, Frati 94a] performs well when the first criterion is considered, with all of the 1500 fits getting within 300cm of the true vertex. This is because bad hits tend to either give complex solutions to the time quadratic or produce quad points far from the main cluster, giving the algorithm an inbuilt method for reducing the effect of bad hits. Its main peak, however, is comparatively spread out, with 7.3% of the fits failing to get within 100cm of the true vertex. The same technique which damps out bad hits also ensures that less of the good hit information is used and so the fitter does not to perform so well under the second criterion.

The Time Fitter, even when given a Quad fitted initial vertex, has a very long tail, with 4.7% of the fits further than 300cm from the true vertex. Its technique of fitting, removing hits with a bad χ^2, refitting etc. is not satisfactory either aesthetically or practically. However, when the fit is good the Time Fitter makes better use of the available statistics (i.e. it uses more of the hits) giving it a sharper peak closer to zero than the Quad Fitter.



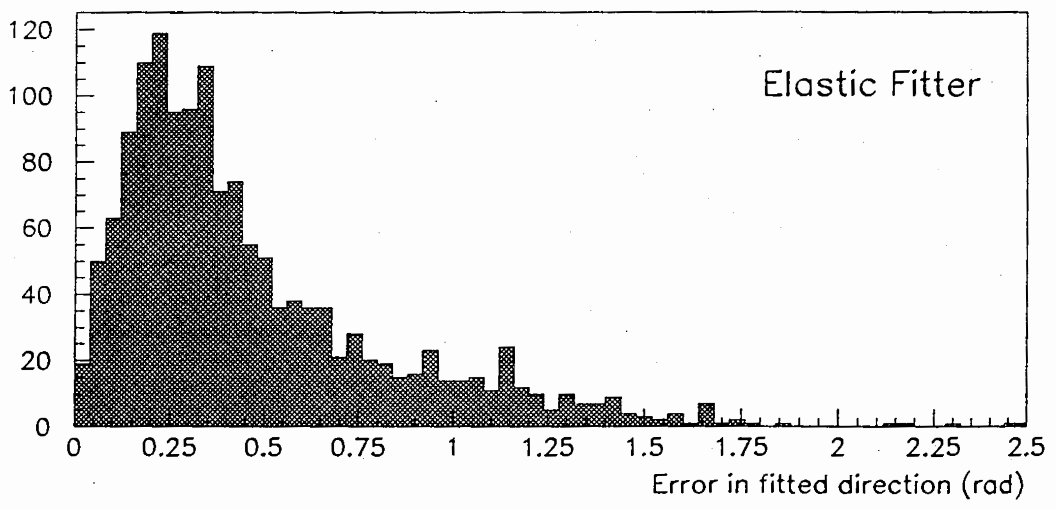


Figure 7: The angle (in radians) between the true and fitted direction for the Time Fitter and the Elastic Fitter.

The Elastic Fitter performs well under both criteria. With its V_i weighting it has good noise rejection and a minimal tail, and with the similarity of the algorithm to a χ^2 fit it has the Time Fitter's sharp peak close to zero.

Figure 7 compares the direction fit results for the Time Fitter (a simple vector sum of the PMT hit directions) and the Elastic Fitter. Here the improvement is quite dramatic, but this should be no surprise given the feeble competition provided by the Time Fitter's direction fitting algorithm.

Whilst the improvement in performance of the Elastic Fitter over both the Quad and Time Fitters is clear it must be admitted that it is far from dramatic. It will probably be concluded that the complexity of the algorithm is not outweighed by quality of the fit. Such judgements should, however, be postponed until tests have been made on a PMT $\beta\gamma$ data set. Such a data set requires considerable CPU time to get enough events over 30 hits, but will be obtained from SNOMAN runs on the HP processor farm at the Rutherford-Appleton Lab. The improved fit quality that the Elastic algorithm provides is still useful for the extraction

Fitter Method	CPU time (in units of the Time time !!)	% of fits outside 100cm	% of fits outside 300cm
Quad	122	7.3%	0.0%
Time	1	8.1%	4.7%
Elastic	4	2.8%	0.0%

Table 1: Timing and fit quality results of the three fitters

of event information for a neural network analysis, and the techniques used by the fitter may provide inspiration for further work.

Shown in Table 1 are the results of timing studies on the three fitters under consideration. It can be seen that the Elastic Fitter has a speed comparable to that of the Time Fitter and considerably faster than that of the Quad Fitter. However, as long as the Elastic algorithm needs an initial Quad fit the speed difference between the two is irrelevant. It is clear that for the Elastic Fitter to be viable an alternative initial fit needs to be found. Care must be taken when making these time comparisons, however, as they are specific to 5MeV electron events. The Time and Elastic fitter CPU times are independant or, at worst, linearly dependent on the number of hits, but the size of the Quad cloud and hence the Quad CPU time depends on the $\sim 4^{\text{th}}$ power of NHITS. Therefore, for event data sets where the mean NHITS is greater than 50 the disparity between the Quad CPU time and the others will be much greater.

5 Comments and Further Developments

It is well known that late hits within the Čerenkov cone tend to push the fit position back along the track direction and such hits outside the cone pull it along the track direction. With more phase space outside the Čerenkov cone then the usual effect of such fitter pull is to drag the fit position along the direction of the electron. It can also be shown [Klein 94] that a vertex determination made by fitting the hits to a Čerenkov cone tends to push the vertex position back along the electron's direction i.e. in the opposite direction to the pull due to hit times. The original thinking behind the Elastic Fitter was to try to use these two opposing pulls to cancel each other by carrying out the position fit with the distance measure M_i^{tot}

$$M_i^{\text{tot}} = M_i^{\text{pos}} + \gamma M_i^{\text{dir}} \tag{16}$$

where γ would be a parameter determining the relative weighting that should be applied to the position and direction components of M_i^{tot}, and M_i^{pos} and M_i^{dir} would have their previous definitions. By carefully chosing the value of γ it was hoped that the two opposing pulls would tend to cancel each other. Unfortunately, and perhaps not surprisingly, this proved not to work. The reason being that compared to the sharp precision of a vertex determination produced using M_i^{pos}, a vertex fit using M_i^{dir} is a very blunt instrument and setting γ large enough to effect a partial cancellation of the pulls meant that the accuracy of the position fit was severely compromised. Although this implementation of pull cancellation proved ineffective the basic idea may warrant further study.

In its original incarnation as the Elastic Arms algorithm [Ohlsson 92, Ohlsson 93] the fitter was constructed to similtaneouly fit multiple tracks in a TPC event. Although modified in this paper, the Elastic Fitter maintains the capability to be simply extended to fit multiple vertices and directions. For the details of how this can be done reference should be made to the two papers cited above. Such an extension may possibly prove useful when fitting neutral current events or even $\beta\gamma$ events.

6 Conclusion

As has already been stated, the quality of fit produced by the Elastic Fitter may not be enough of an improvement to outweigh the compexity of using the algorithm, but it is certainly true that the technique of simulated annealing, the method used to eliminate bad hits, and the possibility of extending the algorithm to handle multiple vertices demand further attention.

References

[Brice 95] Steve Brice. *An Overview of the Feedforward Neural Network Technique and its Application to SNO Event Classification*, 1995. SNO-STR-95-037.

[Frati 94a] Bill Frati. *Quad Fitter Update - I*, 1994.

[Frati 94b] Bill Frati & Richard Van de Water. *Quad Fitter Introduction - Reconstruction and Pattern Recognition*, 1994. SNO-STR-94-030.

[Klein 94] Joshua Klein. *Maximum Likelihood Fitting to Both Time and Angle*, 1994. SNO-STR-94-055.

[Ohlsson 92] Mattias Ohlsson, Carsten Peterson & Alan Yuille. *Track Finding with Deformable Templates – The Elastic Arms Approach.* Computer Physics Communications, vol. 71, No.1–2, pages 77–98, 1992.

[Ohlsson 93] Mattias Ohlsson. *Extensions and Explorations of the Elastic Arms Algorithm.* Computer Physics Communications, vol. 77, No.1, pages 19–32, 1993.