DEUTSCHES ELEKTRONEN-SYNCHROTRON DESY

DESY 71/18
April 1971

MLFIT: A Program to Find Maxima of Likelihood Functions

by

Volker Blobel

2 HAMBURG 52 · NOTKESTIEG 1

MLFIT: A Program to Find Maxima of Likelihood Functions

by

Volker Blobel

The purpose of the FORTRAN IV subroutine MLFIT is to find a local maximum of the logarithm of a likelihood function F (a) of n parameters and to calculate the errors of the parameters. The method is based on the quadratic approximation of the function. The calculation of the function value and of the first and second derivatives of the function with respect to the parameters must be programmed by the user. The number of parameters is not limited by internal dimensions but merely by the total length of a labelled common. It is possible to fix some parameters on given values during the iteration and to perform a special error analysis for some parameters.

CONTENTS

1. Introduction

A common problem in physics is the estimation of parameters $a_1, a_2, \ldots a_n$ of a given function from experimental data. The method of maximum likelihood is often used for the treatment of these problems[1,2]. A likelihood function L ($\underline{a}$) is constructed ($\underline{a} = (a_1, a_2, \ldots a_n)^T$) which gives the relative likelihood that the parameters $a_1, a_2, \ldots a_n$ assume particular values. The maximum likelihood estimator is the value $\underline{a}^*$ which maximizes the likelihood function L ($\underline{a}$) or the logarithm F ($\underline{a}$) = ℓn L($\underline{a}$). The errors σ_j of the parameters a_j can be estimated from the behavior of the function F ($\underline{a}$) in the region near the maximum.

The subroutine MLFIT searches for the maximum of a function F ($\underline{a}$) by carrying out several steps (iterations) in parameter space. The methods used for the recognition of convergence and for the error analysis assume F ($\underline{a}$) to be the logarithm of a likelihood function. Each step carried out in parameter space is calculated from the gradient (vector of the first derivatives) and the matrix of the second derivatives of the function F ($\underline{a}$) with respect to the parameters a_j. Subroutines for the calculation of the derivatives are provided.

If the behavior of the function F ($\underline{a}$) is quadratic, i.e. if the matrix of second derivatives is a constant, one step starting from an arbitrary point may reach the maximum. Under the same condition the error matrix of the parameters equals the inverse of the matrix of the second derivatives.

Likelihood functions F ($\underline{a}$) are often nearly quadratic only near the maximum. In certain regions of parameter space one step may yield a smaller function

value. In such cases one or more shorter steps in a different direction are carried out.

Besides the errors defined by the matrix of the second derivatives other errors can be calculated which depend on the function values in the region near the maximum. This error analysis is time consuming, because additional iterations are necessary.

2. Method

2.1 Search for the maximum

Under certain regularity conditions the function F ($\underline{a}$) can be expanded into a TAYLOR series at the point $\underline{a} = (a_1, a_2, \ldots a_n)^T$.

$$F(\underline{a} + \underline{\Delta a}) = F(\underline{a}) + \underline{g}^T \underline{\Delta a} - \frac{1}{2} \underline{\Delta a}^T G \underline{\Delta a} + \ldots \qquad (1)$$

$$g_\gamma = \frac{\partial F}{\partial a_\gamma} \qquad \text{(gradient)} \qquad (2)$$

$$G_{\gamma\rho} = \frac{\partial^2 F}{\partial a_\gamma \partial a_\rho} \qquad \text{(matrix of the second derivatives)} \qquad (3)$$

Neglecting higher order terms of the expansion the following two basic methods are applicable to find a step in parameter space toward the location of the maximum of the function F ($\underline{a}$).

In the first method the derivatives of the TAYLOR expansion eq. (1) are taken and are set equal to 0. The resulting system of n linear equations is solved for $\underline{\Delta a}_1$.

$$\underline{g} - G \underline{\Delta a}_1 = 0 \qquad (4)$$

$$\underline{\Delta a}_1 = G^{-1} \underline{g} \qquad (5)$$

In the second method a step is carried out in the direction of the gradient $\underline{g}$. Putting $\underline{\Delta a} = t \cdot \underline{g}$ results in a function depending on t. The best value of t can be calculated putting the derivative with respect to t equal to 0.

$$F(\underline{a} + t\,\underline{g}) = F(\underline{a}) + t\;\underline{g}^T\underline{g} - \frac{1}{2}\,t^2\;\underline{g}^T G\,\underline{g} \tag{6}$$

$$\underline{g}^T\underline{g} - t\;\underline{g}^T G\,\underline{g} = 0 \tag{7}$$

$$t = \frac{\underline{g}^T\underline{g}}{\underline{g}^T G\,\underline{g}} \tag{8}$$

$$\underline{\Delta a}_2 = \frac{\underline{g}^T\underline{g}}{\underline{g}^T G\,\underline{g}}\;\underline{g} \tag{9}$$

The methods are illustrated in fig. 1 for the case of two parameters. The first method yields a large step $\underline{\Delta a}_1$ toward the maximum, if the function behaves nearly quadratic. The second method yields a smaller step $\underline{\Delta a}_2$, which gives a better (higher) function value even in the case of strong non-quadratic behavior of the function.

In general several steps in parameter space are necessary to reach the point $\underline{a}^*$, which maximizes the function F ($\underline{a}$).

The method used by MLFIT combines both methods described above. The step $\underline{\Delta a}$ is calculated using a matrix K (with $\varepsilon_1 \approx 0.1$):

$$K = G + \varepsilon_1 \left(2^{\ell} - 1\right) \frac{\underline{g}^T G\,\underline{g}}{\underline{g}^T\underline{g}}\;I \tag{10}$$

(I = unit matrix)

$$\underline{\Delta a} = K^{-1}\,\underline{g} \tag{11}$$

For ℓ=0 K equals G and the method is identical to the first method. For $\varepsilon_1(2^{\ell}-1)\gtrsim 1$ the method is nearly equivalent to the second method. In subroutine MLFIT the parameter ℓ of eq. (10) is modified according to the result of the preceeding steps.

The expected variation of the function in the case of quadratic behavior is ΔF_e.

$$\Delta F_e = \underline{g}^T \underline{\Delta a} - \frac{1}{2} \underline{\Delta a}^T G \underline{\Delta a} \qquad (12)$$

The true variation of the function is ΔF_t.

$$\Delta F_t = F(\underline{a} + \underline{\Delta a}) - F(\underline{a}) \qquad (13)$$

In addition a value ΔF_c is defined, which is used for the comparison of different function values (with $\varepsilon_2 \approx 0.1$, $\varepsilon_3 \approx 10^{-5}$).

$$\Delta F_c = \max\left(\varepsilon_2 , \varepsilon_3 \cdot |F(\underline{a} + \underline{\Delta a})|\right) \qquad (14)$$

The strategy of the program depends in the following manner on the result of the preceeding step, in which $\underline{a}$ was replaced by $\underline{a} + \underline{\Delta a}$ (with $\varepsilon_4 \approx 0.5$, $\ell_d \approx 2$).

Classification of step	Condition	Action
I. Divergent	$\Delta F_t + \Delta F_c < 0$	ℓ is replaced by $\ell+\ell_d$; parameter values are replaced by preceeding values.
II. Slowly convergent	$\Delta F_t < \varepsilon_4 \cdot \Delta F_e$ and $\Delta F_t > \Delta F_c$	ℓ is replaced by $\ell+1$
III. Convergent	$\Delta F_t \geq \varepsilon_4 \cdot \Delta F_e$ or $\Delta F_t \leq \Delta F_c$	ℓ is replaced by max. $(\ell-1,0)$

The next step is calculated according to eq. (10) using the corresponding values of $\underline{g}$ and G and the new value of ℓ. Final convergence is assumed, if the following conditions are satisfied:

1. $\Delta F_e < \Delta F_c$
2. At least two iterations are performed
3. $\ell = 0$

If these conditions are satisfied and $\Delta F_c \leq \frac{1}{2}$, the last step lies within the error ellipsoid of the quadratic approximation eq. (1) for one standard deviation. In this region of parameter space the function behavior is assumed to be nearly quadratic and the last step carried out with $\ell = 0$ should reach the point $\underline{a}^*$ in parameter space, which maximizes the function F ($\underline{a}$).

Due to some physical constraints a certain region of parameter space may be forbidden. If a step is carried out into the forbidden region, the program simply treats this step as divergent.

2.2 Calculation of errors

The error matrix $E = (E_{jk})$ of the parameters is found by inversion of the matrix G of second derivatives. The error σ_j of the paramter a_j is the square root of the corresponding diagonal element.

$$E = G^{-1} \tag{15}$$

$$\sigma_j = (E_{jj})^{1/2} \tag{16}$$

These errors are good estimates if the function is nearly quadratic near the maximum. They are symmetric by definition.

For a discussion of the error definition a function $F_j(\underline{a}, \alpha)$ is defined:

$$F_j(\underline{a}, \alpha) = \mathrm{Max}\left\{ F(\underline{a}) \; ; \; a_j = a_j^* + \alpha \right\} \tag{17}$$

$F_j(\underline{a},\alpha)$ is the maximum function value under the constraint $a_j = a_j^* + \alpha$. The following relation is valid for a one standard deviation error σ_j:

$$F(\underline{a}^*) - F_j(\underline{a}, \sigma_j) = \frac{1}{2} \tag{18}$$

For an error s_j corresponding to k standard deviations:

$$F(\underline{a}^*) - F_j(\underline{a}, s_j) = \frac{1}{2} k^2 \tag{19}$$

In the subroutine MLFIT the error analysis for a parameter a_j is performed in the following way:

The parameter a_j is modified by $k = \pm\varepsilon_5$ symmetric standard deviations (e.g. ε_5=1.0) and according to eq. (17) the maximum of the function $F(\underline{a})$ with respect to the other parameters is determined. Using this function

value a new error s_j is calculated according to eq. (19) by using the following second quadratic approximation:

$$S_j = k\sigma_j \left(\frac{k^2}{2\left(F(\underline{a}^*) - F_j(\underline{a}, k\sigma_j)\right)} \right)^{1/2} \qquad (20)$$

These errors may be asymmetric. The geometrical meaning of eq. (20) is illustrated in fig. 2.

Because of the difference between two function values these errors may be subject to large rounding errors. To get a more precise value for $F(\underline{a}^*)$ an additional iteration is performed after convergence.

3. Usage

3.1 Formal rules

A program that uses MLFIT may be divided into three parts. The first part (A) is the initialization of MLFIT. The other two parts are executed once per iteration; in the second part (B) the function $F(\underline{a})$ and the derivatives are calculated; in the third part (C) the results of the last iteration are tested, and a new step is calculated. Execution of part B and C is repeated, until either convergence is reached or a given number of iterations is performed.

The communication between the user program and the subroutine MLFIT is partly via arguments, partly via the common PARCOM.

COMMON / PARCOM / NP, NE, NF, NW, FCT, PA (dim)

The length of the array PA must be dim $\geq$ ndim,

ndim = $3(n^2 + 5n)/2$ for n parameters[3]. In MLFIT the length is dim = 75 corresponding to n = 5. All the vectors and matrices (in symmetric storage mode) are stored in the array PA.

			During iteration	After convergence
PA (1)	...	PA (n)	parameter $\underline{a}$	parameter $\underline{a}^*$
PA (n+1)	...	PA (2n)	gradient $\underline{g}$	errors $\underline{\sigma}$
PA (2n+1)	...	PA (2n+n(n+1)/2)	matrix G	error matrix E

The remaining part of the array PA is used internally.

Part A

The subroutine MLFIT is initialized by the call:

CALL MLFIT (NPR, ITR, IAD, IPR)

NPR = number n of parameters

ITR = maximum number of iterations

IAD = 1 Part B is executed finally (additional) using the best values of the parameters (in case of convergence or non-convergence)

= 0 No additional execution of part B

IPR = 0 No print-out

= 1 Print-out of results

= 2 Print-out of iterations and results

The number of parameters is stored in NP by MLFIT. The other variables of common PARCOM are explained in Part B. The initial values of the parameters may be defined before or after the initialization by the user program. After initialization of MLFIT the user program may use the following options:

1. By calling FIXPA (j) the parameter a_j is kept fixed on the given value during the iteration.

2. By calling ERRPA (k) an internal flag is set, and after convergence an error analysis is performed for parameter a_k.

3. Constants internal to MLFIT may be modified (see Chapter 4).

Part B

The values of the function F ($\underline{a}$) and the first and second derivatives are to be calculated by the user program and stored in the common PARCOM.

Function value:	FCT	$= F(\underline{a})$
gradient $\underline{g}$:	PA (n+j)	$= \frac{\partial F}{\partial a_j}$
matrix G:	PA (2n+j+k(k-1)/2)	$= -\frac{\partial^2 F}{\partial a_j \partial a_k}$ $\quad j \leq k$

Special conditions are to be flagged by the user by defining the flag NF.

NF = 1 step into a forbidden region

NF = 2 parameter (s) modified by user program

The number of terms used in the calculation of the function may be counted in NE. NE and NW are not used by MLFIT.

The variables NE, NF, NW, FCT, PA (n+1) ... PA (2 n + n (n+1)/2) are set to 0 by MLFIT before part B is entered.

Part C

One iteration is performed by calling:

CALL MALIT ($\&s_1$, $\&s_2$)

Depending on some tests the program branches by using different returns to the calling program.

Normal return	convergence reached
return 1 (jump to statement no. S_1)	further iteration required
return 2 (jump to statement no. S_2)	non-convergence after ITR iterations

At normal return the results (parameters $\underline{a}^*$, errors $\underline{\sigma}$ and error matrix E) are stored in array PA. If return 1 is used, the next values of the parameters to be used in part B are stored in PA (1) ..... PA (n). At return 2 (non-convergence) the best values of the parameters obtained so far are stored in PA (1) ... PA (n).

The formal rules are summarized in table 1.

3.2 Function evaluation

Three subroutines are provided for the calculation of the function F ($\underline{a}$) and the first and second derivatives of the function. They are applicable in three different problems of parameter estimation, described below.

1. Method of least squares

An experiment consists of N independent measurements y_i, i=1 ... N, of some quantity y at coordinates x_i. The errors of the y_i are dy_i. The values y_i are to be fitted to a function $f(x_i, \underline{a})$ dependent on n parameters a_1 ... a_n.

According to the method of least squares the best fit is defined by the minimum of the weighted sum of squares:

$$S(\underline{a}) = \sum_{i=1}^{N} w_i \cdot \left(f(x_i, \underline{a}) - y_i \right)^2 \tag{21}$$

$$w_i = \left(dy_i \right)^{-2} \tag{22}$$

In the case of normal distributed errors dy_i the logarithm of the likelihood function F ($\underline{a}$) for the problem equals $-\frac{1}{2}$ times the function S ($\underline{a}$) plus a constant term:

$$F\ (\underline{a}) = -\frac{1}{2}\ S\ (\underline{a}) + \text{const.} \tag{23}$$

Omitting the constant term the following equations are obtained for the function F ($\underline{a}$) and the derivatives:

$$F(\underline{a}) = -\frac{1}{2} \sum_i w_i \cdot \left(f(x_i, \underline{a}) - y_i \right)^2 \tag{24}$$

$$\frac{\partial F}{\partial a_j} = \sum_i w_i \frac{\partial f}{\partial a_j} \left(y_i - f(x_i, \underline{a}) \right) \tag{25}$$

$$-\frac{\partial^2 F}{\partial a_j \partial a_k} = \sum_i w_i \cdot \left(\frac{\partial f}{\partial a_j} \frac{\partial f}{\partial a_k} - \frac{\partial^2 f}{\partial a_j \partial a_k} \left(y_i - f(x_i, \underline{a}) \right) \right) \tag{26}$$

2. Poisson distribution

An experiment consists of N independent measurements y_i, i = 1 ... N, of some quantity y at coordinates x_i. The values y_i are poisson distributed, e.g. they represent counting rates (numbers of events) in a small region of x. The expected counting rate be f (x_i, $\underline{a}$) depending on n parameters $a_1 \ldots a_n$.

Problems of this kind are often treated by the method of least squares by associating an error $dy_i = (y_i)^{1/2}$ or $(f(x_i, \underline{a}))^{1/2}$ to each y_i. Difficulties arising in this method, if some y_i are small or even zero can be avoided by using the poisson distribution in the method of maximum likelihood.

The probability of observing a counting rate y_i in a region x_i expecting a counting rate $f(x_i, \underline{a})$ is given by the following expression:

$$P(x_i, \underline{a}) = \frac{f(x_i, \underline{a})^{y_i}}{y_i!} \exp\left(-f(x_i, \underline{a})\right) \tag{27}$$

The likelihood function $L(\underline{a})$ is the product of all probabilities $P(x_i, \underline{a})$.

$$L(\underline{a}) = \prod_i \frac{f(x_i, \underline{a})^{y_i}}{y_i!} \exp\left(-f(x_i, \underline{a})\right) \tag{28}$$

The following equations are obtained for the logarithm $F(\underline{a}) = \ln L(\underline{a})$ and the derivatives:

$$F(\underline{a}) = \sum_i y_i \ln f(x_i, \underline{a}) - \sum_i f(x_i, \underline{a}) + \text{const} \tag{29}$$

$$\frac{\partial F}{\partial a_j} = \sum_i y_i \frac{\frac{\partial f}{\partial a_j}}{f(x_i, \underline{a})} - \sum_i \frac{\partial f}{\partial a_j} \tag{30}$$

$$-\frac{\partial^2 F}{\partial a_j \partial a_k} = \sum_i y_i \frac{\frac{\partial f}{\partial a_j}\frac{\partial f}{\partial a_k} - \frac{\partial^2 f}{\partial a_j \partial a_k} f(x_i, \underline{a})}{f^2(x_i, \underline{a})} - \sum_i \frac{\partial^2 f}{\partial a_j \partial a_k} \tag{31}$$

3. Method of maximum likelihood

An experiment consists of N independent observations (events) at coordinates x_i, i=1 ... N. The expected distribution of the observations (probability density) is given by a function $f(x_i, \underline{a})$ depending on n parameters $a_1 \ldots a_n$.

First f (x_i, $\underline{a}$) is assumed to be normalized to unity (X=range of observation):

$$\int_X f(x, \underline{a})\, dx = 1 \tag{32}$$

The likelihood function for the problem is given by the following product:

$$L(\underline{a}) = \prod_i f(x_i, \underline{a}) \tag{33}$$

The following equations are obtained for the logarithm F ($\underline{a}$) = ℓn L ($\underline{a}$) and the derivatives:

$$F(\underline{a}) = \sum_i \ell n\, f(x_i, \underline{a}) \tag{34}$$

$$\frac{\partial F}{\partial a_j} = \sum_i \frac{\frac{\partial f}{\partial a_j}}{f(x_i, \underline{a})} \tag{35}$$

$$-\frac{\partial^2 F}{\partial a_j \partial a_k} = \sum_i \frac{\frac{\partial f}{\partial a_j}\frac{\partial f}{\partial a_k} - \frac{\partial^2 f}{\partial a_j \partial a_k} f(x_i, \underline{a})}{f^2(x_i, \underline{a})} \tag{36}$$

Often the probability density is not normalized to unity but the integral eq. (32) equals the expected number of observations (events) $\bar{N}(\underline{a})$.

$$\int_X f(x, \underline{a})\, dx = \bar{N}(\underline{a}) \tag{37}$$

Then the extended maximum likelihood function[2] is to be used, which contains an additional factor representing the probability of observing N events, if $\bar{N}(\underline{a})$ are expected.

$$L(\underline{a}) = \frac{\exp\left(-\int_X f(x, \underline{a})\, dx\right)}{N!} \prod_i f(x_i, \underline{a}) \tag{38}$$

The following equations are obtained for the logarithm $F(\underline{a}) = \ln L(\underline{a})$ and the derivatives:

$$F(\underline{a}) = \sum_i \ln f(x_i, \underline{a}) - \int_X f(x, \underline{a})\, dx \tag{39}$$

$$\frac{\partial F}{\partial a_\gamma} = \sum_i \frac{\frac{\partial f}{\partial a_\gamma}}{f(x_i, \underline{a})} - \int_X \frac{\partial f}{\partial a_\gamma}\, dx \tag{40}$$

$$-\frac{\partial^2 F}{\partial a_\gamma \partial a_\kappa} = \sum_i \frac{\frac{\partial f}{\partial a_\gamma}\frac{\partial f}{\partial a_\kappa} - \frac{\partial^2 f}{\partial a_\gamma \partial a_\kappa} f(x_i, \underline{a})}{f^2(x_i, \underline{a})} + \int_X \frac{\partial^2 f}{\partial a_\gamma \partial a_\kappa}\, dx \tag{41}$$

The equations (26), (31), (36) and (41) for the second derivatives of $F(\underline{a})$ contain the second derivatives of the functions f (x, $\underline{a}$). Apart from the labour of programming and computing these derivatives there are advantages in neglecting these second derivatives of f (x, $\underline{a}$) with respect to $\underline{a}$[4]. The position of the maximum of the function F ($\underline{a}$) is not influenced by this approximation, although the errors of the parameters defined by the inverse of the matrix of second derivatives of F ($\underline{a}$) are influenced. The change in the errors, which in general will be small can be checked by performing the error analysis, which uses the values of the function F ($\underline{a}$) itself in the region near the maximum.

Neglecting the second derivatives of function f (x, $\underline{a}$) the summation necessary for the calculation of F ($\underline{a}$) and the derivatives of F ($\underline{a}$) can be done with the subroutines LESQU, POISS and MLIKE, respectively, for the three problems.

The common EQUAT serves for the transmission of data to the subroutines.

COMMON / EQUAT / EV, FV, FD, WT, A (dim)

The length of array A must be dim $\geq$ n for n parameters.[3] In the subroutines the length is dim = 5. In a loop on index i the values corresponding to a single measurement (y_i, $f(x_i, \underline{a})$ etc.) must be stored in common EQUAT and the appropriate subroutine must be called. The meaning of the variables in common EQUAT and the calculations performed by the subroutines are given in table 2.

4. Program structure

The subroutine MLFIT has additional entries MALIT, ERRPA and FIXPA. The subroutines GMAT and PFIT are called by MLFIT. The subroutines LESQU, POISS and MLIKE are independent of all other subroutines. All working areas except local indices are contained in the commons PARCOM (see Ch. 3.1) and MALARR.

COMMON/MALARR/FPA (20), NPA (20), NN(9)

The meaning of the variables in common MALARR is given in table 3. Relevant to the user are especially those words, which are initialized to certain values. These values are given in parentheses; they can be modified by the user after the call of MLFIT.

After return from MALIT the variable NPA (3) contains the status of the program. The meaning is:

NPA (3) = 0 convergence reached

= 1 further iteration necessary

= 2 non-convergence

If NPA (4) = j≠0, the error analysis for parameter a_j is just completed, the results of the error analysis are contained in FPA(I), NPA(I), I = 17 ... 20. A frequent change of the fitparameter ℓ (see eq. (10)) between 0 and larger values during the iteration indicates a strong non-quadratic behavior of the function F (a). In such cases it may be useful to start with ℓ>0 (e.g. NPA (1)=2), or to increase ℓ_d and decrease ε_1 simultaneously (e.g. NPA (8)=4, FPA (1) = 0.05). If the accuracy of the function calculation is limited by large rounding errors, it may be useful to increase ε_2 and ε_3 (see eq. (14)).

Subroutine GMAT:

All the matrix operations of eqs. (10), (11) and (12) are performed in subroutine GMAT. If the condition of matrix K (eq. (10)) is very bad, e.g. if two parameters are strongly correlated, some elements of the matrix may become very small during the process of matrix inversion. To avoid absurd results due to rounding errors, the matrix is only partly inverted, if certain matrix elements decrease by more than a factor of TOL (TOL ≈ 10^{-4}). The parameter corresponding to this matrix element is not modified during the current iteration, and the corresponding error is not defined, i.e. the parameter is treated like a fixed parameter.

Subroutine PFIT (IGØ):

All print-out is done in the subroutine PFIT.

IGØ = 1 call after one iteration

= 2 call after the last iteration, if convergence is reached

= 3 call after completion of the error analysis for one parameter

= 4 final call before the last return from MALIT

The print-out is explained in table 4.

5. Example

The example given below demonstrates the use of subroutine MLFIT and subroutines LESQU, POISS and MLIKE.

In the main program N measurements t_i in the region $t_a < t_i < t_b$ are faked according to the probability density $\sim \exp(-b \cdot t)$, using random numbers (random number generator ZPF). The N values t_i are related to NB t-regions of width Δt, in order to fake counting rates y_i for NB values $\bar{t}_i$.

The parameters a and b of the model $dN/dt = a \cdot \exp(-b \cdot t)$ are estimated by three different methods.

The first and the second method (subroutine XPBFIT) use the counting rates y_i. The expected counting rate at $\bar{t}_i$ is given by

$$\int_{\bar{t}_i - \Delta t/2}^{\bar{t}_i + \Delta t/2} a \cdot \exp(-bt)\, dt$$

which is approximated by

$$f(\bar{t}_i, a, b) = \Delta t \cdot a \exp(-b\bar{t}_i)$$

The first method, the method of least squares, uses an error $dy_i = (y_i)^{1/2}$ associated to each counting rate y_i; values y_i, $i \geq k$ are not used, if $y_k < 5$. The second method is based on the poisson distribution for each value y_i; all values y_i are used. The only difference in the program is the use of subroutine LESQU in the first method and the use of subroutine POISS in the second method.

The third method is the extended maximum likelihood method, using the N values t_i. The integrals necessary in this method are calculated in subroutine XPMFIT, which in addition calls subroutine MLIKE.

The results of the second and the third method are nearly identical. The computer time for the third method is much higher, however, because of the greater number of terms used. The first method yields slightly different results and larger errors for the parameters, partly due to the smaller t-region. The error analysis indicates a sufficient accuracy of the symmetric errors.

```
      COMMON/PARCOM/ NP,NE,NF,NW,FCT,PA(75)
      COMMON/ZUF/ IX
      REAL T(2000),TZ(120),YZ(120)
      REAL A(4),B(4),C(4),R(4)
      IX=842919591
C     -----
      N=2000
      B(1)=10.0
      TA=0.01
      TB=0.5
      NB=49
C     -----
      EBTA=EXP(-B(1)*TA)
      EBTB=EXP(-B(1)*TB)
      DAB=EBTA-EBTB
      A(1)=FLOAT(N)*B(1)/DAB
      DO 10 I=1,N
    2 T(I)=-ALOG(EBTA-ZPF(DUMMY)*DAB)/B(1)
      IF(T(I).LE.TA.OR.T(I).GE.TB) GOTO 2
   10 CONTINUE
C
      BIN=(TB-TA)/FLOAT(NB)
      TZ(1)=TA+0.5*BIN
      YZ(1)=0.0
      DO 12 J=2,NB
      TZ(J)=TZ(J-1)+BIN
   12 YZ(J)=0.0
      DO 14 I=1,N
      J=(T(I)-TA)/BIN+1.0
   14 YZ(J)=YZ(J)+1.0
C
      CALL XPBFIT(NB,TZ,YZ,1)
      A(2)=PA(1)
      B(2)=PA(2)
C
      CALL XPBFIT(NB,TZ,YZ,2)
      A(3)=PA(1)
      B(3)=PA(2)
C
      CALL XPMFIT(N,TA,TB,T)
      A(4)=PA(1)
      B(4)=PA(2)
C
      WRITE(6,101)
      TV=TA
      DO 18 K=1,4
      IF(A(K).EQ.0.0) GOTO 18
      C(K)=A(K)*EXP(-B(K)*TV)
   18 CONTINUE
      DO 30 J=1,NB
      TV=TV+BIN
      DO 20 K=1,4
      R(K)=0.0
      IF(A(K).EQ.0.0) GOTO 20
      H=C(K)
      C(K)=A(K)*EXP(-B(K)*TV)
      R(K)=(H-C(K))/B(K)
   20 CONTINUE
      WRITE(6,102) TZ(J),YZ(J),R
   30 CONTINUE
      STOP
  101 FORMAT(1H1,11X,'T',8X,'Y',6X,'THEORY',5X,'LSQ-FIT',
     1  1X,'POISSON-FIT',6X,'ML-FIT'/)
  102 FORMAT(1X,F15.3,F6.0,4F12.2)
      END
```

```
      SUBROUTINE XPBFIT(NB,TZ,YZ,M)
      COMMON/PARCOM/ NP,NE,NF,NW,FCT,PA(75)
      COMMON/EQUAT/  EV,FV,FD,WT,A(5)
      REAL TZ(1),YZ(1)
      BIN=TZ(2)-TZ(1)
      IF(M.EQ.1) GOTO 10
      IF(M.EQ.2) GOTO 20
      GOTO 60
C
   10 ASSIGN 42 TO NFIT
      WRITE(6,101)
      DO 12 I=1,NB
      IF(YZ(I).LT.5.0) GOTO 14
   12 CONTINUE
      I=NB+1
   14 II=I-1
      GOTO 30
C
   20 ASSIGN 44 TO NFIT
      WRITE (6,102)
      II=NB
C     ESTIMATE INITIAL VALUES
   30 IF(YZ(1).LE.0.0) GOTO 60
      K=II/2
   32 K=K-1
      IF(K.LT.3) GOTO 60
      IF(YZ(K).LE.0.0) GOTO 32
      PA(2)=(ALOG(YZ(K))-ALOG(YZ(1)))/(TZ(1)-TZ(K))
      PA(1)=YZ(1)*EXP(PA(2)*TZ(1))/BIN
C
C     PART A
      CALL MLFIT(2,6,0,2)
      CALL ERRPA(1)
      CALL ERRPA(2)
C
C     PART B
   40 DO 50 I=1,II
      EV=YZ(I)
      DV=EXP(-PA(2)*TZ(I))*BIN
      FV=PA(1)*DV
      A(1)=DV
      A(2)=-TZ(I)*PA(1)*DV
      GOTO NFIT,(42,44)
   42 WT=1.0/YZ(I)
      CALL LESQU
      GOTO 50
   44 CALL POISS
   50 CONTINUE
C
C     PART C
      CALL MALIT(&40,&60)
      GOTO 100
C
   60 PA(1)=0.0
      PA(2)=0.0
  100 RETURN
  101 FORMAT(/'1 LEAST SQUARE FIT'/)
  102 FORMAT(/'1 POISSON FIT'/)
      END
```

```
      SUBROUTINE XPMFIT(N,TA,TB,T)
      COMMON/PARCOM/ NP,NE,NF,NW,FCT,PA(75)
      COMMON/EQUAT/  EV,FV,FD,WT,A(5)
      REAL T(1)
      IF(PA(1).EQ.0.0.OR.PA(2).EQ.0.0) GOTO 30
      WRITE(6,101)
C
C     PART A
      CALL MLFIT(2,6,0,2)
      CALL ERRPA(1)
      CALL ERRPA(2)
C
C     PART B
   10 EBTA=EXP(-PA(2)*TA)
      EBTB=EXP(-PA(2)*TB)
      DAB=EBTA-EBTB
      FCT=-PA(1)*DAB/PA(2)
      PA(NP+1)=-DAB/PA(2)
      PA(NP+2)=-PA(1)*(TB*EBTB-TA*EBTA-DAB/PA(2))/PA(2)
      DO 20 I=1,N
      DV=EXP(-PA(2)*T(I))
      FV=PA(1)*DV
      A(1)=DV
      A(2)=-PA(1)*T(I)*DV
   20 CALL MLIKE
C
C     PART C
      CALL MALIT(&10,&30)
      GOTO 100
   30 PA(1)=0.0
      PA(2)=0.0
  100 RETURN
  101 FORMAT(/'1 EXT.MAXIMUM LIKELIHOOD FIT'/)
      END
```

LEAST SQUARE FIT

ITERATIONS

IT	FCT	EST.FCT	NE	L	X	F	PA . . .	
0	-18.0933		37			0	22063.9	9.61690
1	-14.4981	-14.5219	37	0	0	0	22276.5	10.1019
2	-14.4841	-14.4844	37	0	0	0	22339.1	10.1453
3		-14.4841		0	0		22341.4	10.1464

ERROR ANALYSIS ERROR = 1.000 SD

I	PA(I)	SYM ERROR	ESTDFCT	DFCT(+)	ERROR(+)	NFCT	X	DFCT(-)	ERROR(-)	NFCT	X
1	22341.4	817.713	0.500	0.484539	830.654	2	1	0.502509	-815.670	2	1
2	10.1464	0.287851	0.500	0.476391	0.294897	2	1	0.502403	-0.287162	2	1

CONVERGENCE FCT= -14.4841 NFCT= 3
NE= 37 F= 0

I	PA(I)	SYM.SD	CORR
1	22341.4	817.713	
2	10.1464	0.287851	0.78

POISSON FIT

ITERATIONS

IT	FCT	EST.FCT	NE	L	X	F	PA . . .	
0	6673.00		49			0	22065.5	9.62174
1	6674.84	6674.93	49	0	0	0	22153.6	9.95597
2	6674.84	6674.84	49	0	0	0	22142.5	9.94258
3		6674.84		0	0		22142.1	9.94240

ERROR ANALYSIS ERROR = 1.000 SD

I	PA(I)	SYM ERROR	ESTDFCT	DFCT(+)	ERROR(+)	NFCT	X	DFCT(-)	ERROR(-)	NFCT	X
1	22142.1	762.323	0.500	0.484375	774.518	2	1	0.511719	-753.544	2	1
2	9.94240	0.244930	0.500	0.492188	0.246866	2	1	0.500000	-0.244930	2	1

CONVERGENCE FCT= 6674.84 NFCT= 3
NE= 49 F= 0

I	PA(I)	SYM.SD	CORR
1	22142.1	762.323	
2	9.94240	0.244930	0.76

EXT.MAXIMUM LIKELIHOOD FIT

ITERATIONS

IT	FCT	EST.FCT	NE	L	X	F	PA . . .	
0	15887.4		2000			0	22142.1	9.94240
1	15887.4	15887.4	2000	0	0	0	22159.4	9.95212
2	15887.4	15887.4	2000	0	0	0	22170.8	9.95495
3		15887.4		0	0		22158.5	9.95196

ERROR ANALYSIS ERROR = 1.000 SD

I	PA(I)	SYM ERROR	ESTDFCT	DFCT(+)	ERROR(+)	NFCT	X	DFCT(-)	ERROR(-)	NFCT	X
1	22158.5	762.624	0.500	0.488281	771.718	2	1	0.503906	-759.663	2	1
2	9.95196	0.244868	0.500	0.492188	0.246803	2	1	0.496094	-0.245830	2	1

CONVERGENCE FCT= 15887.4 NFCT= 3
NE= 2000 F= 0

I	PA(I)	SYM.SD	CORR
1	22158.5	762.624	
2	9.95196	0.244868	0.76

T	Y	THEORY	LSQ-FIT	POISSON-FIT	ML-FIT
0.015	191.	191.75	191.95	190.82	190.94
0.025	180.	173.51	173.43	172.76	172.85
0.035	157.	156.99	156.70	156.41	156.48
0.045	141.	142.05	141.58	141.61	141.65
0.055	125.	128.54	127.92	128.21	128.23
0.065	120.	116.30	115.58	116.07	116.09
0.075	99.	105.24	104.43	105.09	105.09
0.085	103.	95.22	94.35	95.14	95.14
0.095	79.	86.16	85.25	86.14	86.12
0.105	75.	77.96	77.02	77.99	77.96
0.115	74.	70.54	69.59	70.60	70.58
0.125	66.	63.83	62.88	63.92	63.89
0.135	50.	57.76	56.81	57.87	57.84
0.145	47.	52.26	51.33	52.40	52.36
0.155	45.	47.29	46.38	47.44	47.40
0.165	54.	42.79	41.90	42.95	42.91
0.175	41.	38.71	37.86	38.88	38.85
0.185	34.	35.03	34.21	35.20	35.17
0.195	36.	31.70	30.90	31.87	31.84
0.205	26.	28.68	27.92	28.85	28.82
0.215	18.	25.95	25.23	26.12	26.09
0.225	24.	23.48	22.79	23.65	23.62
0.235	23.	21.25	20.60	21.41	21.38
0.245	12.	19.22	18.61	19.39	19.36
0.255	21.	17.40	16.81	17.55	17.52
0.265	13.	15.74	15.19	15.89	15.86
0.275	15.	14.24	13.72	14.39	14.36
0.285	14.	12.89	12.40	13.03	13.00
0.295	16.	11.66	11.20	11.79	11.77
0.305	9.	10.55	10.12	10.68	10.65
0.315	8.	9.55	9.15	9.67	9.64
0.325	5.	8.64	8.26	8.75	8.73
0.335	12.	7.82	7.47	7.92	7.90
0.345	14.	7.07	6.75	7.17	7.15
0.355	7.	6.40	6.10	6.49	6.48
0.365	12.	5.79	5.51	5.88	5.86
0.375	5.	5.24	4.98	5.32	5.31
0.385	3.	4.74	4.50	4.82	4.81
0.395	4.	4.29	4.06	4.36	4.35
0.405	3.	3.88	3.67	3.95	3.94
0.415	1.	3.51	3.32	3.58	3.57
0.425	4.	3.18	3.00	3.24	3.23
0.435	3.	2.88	2.71	2.93	2.92
0.445	1.	2.60	2.45	2.65	2.64
0.455	1.	2.35	2.21	2.40	2.39
0.465	3.	2.13	2.00	2.18	2.17
0.475	3.	1.93	1.80	1.97	1.96
0.485	1.	1.74	1.63	1.78	1.78
0.495	2.	1.58	1.47	1.61	1.61

References and Footnotes:

1. M.G. Kendall and A. Stuart, The advanced theory of statistics, vol. 2 (Griffin, London, 1961) 35

2. W.P. Swanson, DESY-Report 66/17

3. The way in which labelled commons of different lengths are handled by the computer system must be taken into consideration by the user.

4. E.M.L. Beale, Nonlinear programming (ed. J. Abadie), (North Holland, Amsterdam, 1967) 138.

Part A

dim $\geq 3\ (n^2 + 5n)/2$	COMMON/PARCOM/NP, NE, NF, NW, FCT, PA (dim) (COMMON/MALARR/ FPA(20), NPA(20), NN(9))
NPR = number n of parameters	CALL MLFIT (NPR, ITR, IAD, IPR)
ITR = max. number of iter.	
IAD = 1 additional exec. of part B	
= 0 no add. exec.	
IPR = 0 no print-out	
= 1 print-out of result	
= 2 print-out of iter. and result	
definition of initial values	PA (1) = . . . PA (n) =
options:	
param. a_j fixed	CALL FIXPA (j)
error analysis for parameter a_k	CALL ERRPA (k)
modification of constants (COMMON MALARR necessary)	FPA ( ) = NPA ( ) =

Part B

function value	S_1	FCT =
first derivatives		PA (n+j) =
second derivatives		PA (2n + j + k (k-1)/2) =
number of terms		NE =
if special conditions		NF = 1 or 2

Part C

		CALL MALIT (& S_1, & S_2)
convergence reached		...
non-convergence	S_2	...

Table 1 : Formal rules for usage

Table 2: Input and results of the subroutines LESQU, POISS and MLIKE

		LESQU	POISS	MLIKE
Input:	EV	y_i	y_i	—
	FV	$f(x_i, \underline{a})$	$f(x_i, \underline{a})$	$f(x_i, \underline{a})$
	FD	—	—	—
$i = 1 \ldots N$	WT	$\omega_i = (dy_i)^{-2}$	—	—
	A(j), j=1...NP	$\frac{\partial f}{\partial a_j}$	$\frac{\partial f}{\partial a_j}$	$\frac{\partial f}{\partial a_j}$
Calculation:				
FCT		$-\frac{1}{2}\sum_i \omega_i \left(f(x_i, \underline{a}) - y_i\right)^2$	$\sum_i y_i \ln f(x_i, \underline{a})$	$\sum_i \ln f(x_i, \underline{a})$
PA (NP + j)		$\sum_i \omega_i \frac{\partial f}{\partial a_j}\left(y_i - f(x_i, \underline{a})\right)$	$\sum_i y_i \frac{\frac{\partial f}{\partial a_j}}{f(x_i, \underline{a})} - \sum_i \frac{\partial f}{\partial a_j}$	$\sum_i \frac{\frac{\partial f}{\partial a_j}}{f(x_i, \underline{a})}$
PA (2 NP + j + k (k-1)/2) $j \leq k$		$\sum_i \omega_i \frac{\partial f}{\partial a_j} \frac{\partial f}{\partial a_k}$	$\sum_i y_i \frac{\frac{\partial f}{\partial a_j} \frac{\partial f}{\partial a_k}}{f^2(x_i, \underline{a})}$	$\sum_i \frac{\frac{\partial f}{\partial a_j} \frac{\partial f}{\partial a_k}}{f^2(x_i, \underline{a})}$
NE		$\sum_i 1 = N$	$\sum_i 1 = N$	$\sum_i 1 = N$
NF			$=1$, if $f(x_i, \underline{a}) \leq 0$	$=1$, if $f(x_i, \underline{a}) \leq 0$

COMMON/MALARR/FPA (20), NPA (20), NN (9)

I	FPA (I)	NPA (I)
1	ε_1 (0.1)	fit parameter ℓ (0)
2	ε_2 (0.1)	max. no. of it. in error analysis (4)
3	ε_3 (10^{-5})	status
4	ε_4 (0.5)	index j
5	ε_5 (1.0)	= IPR (argument)
6	ΔF_c	= IAD (argument)
7	ΔF_t	flag
8	working area	ℓd (2)
9		number NE (*)
10		no. of fixed parameters (*)
11	F ($\underline{a}^*$)	no. of iterations (*)
12	Max F($\underline{a}$)	no. of iterations
13	F($\underline{a}$)	flag for convergence
14	F($\underline{a}$) + ΔF_e	max. no. of iterations
15	ΔF_e	no. of fixed parameters
16	k= $\pm \varepsilon_5$	index j
17	F($\underline{a}^{\neq}$) - Fj ($\underline{a}$, $\varepsilon_5 \cdot \sigma_j$)	no. of iterations (+)
18	S_j(+)	no. of fixed parameters (+)
19	F($\underline{a}^{\neq}$) - F_j($\underline{a}$, $-\varepsilon_5 \cdot \sigma_j$)	no. of iterations (-)
20	S_j(-)	no. of fixed parameters (-)

Table 3: Content of common MALARR. Values marked by (*) are final values after convergence

For each iteration the following is printed:

IT	=	no. of iteration
FCT	=	F ($\underline{a}$)
EST.FCT	=	F ($\underline{a}$) + ΔF_e (previous iteration)
NE	=	number NE
L	=	fit parameter ℓ
X	=	no. of fixed parameters
F	=	flag NF
PA...	=	parameters $\underline{a}$

For each error analysis the factor ε_5 and the following is printed:

I	=	index of parameter
PA (I)	=	a_i^*
SYM. ERROR	=	$\varepsilon_5 \cdot \sigma_i$ (symmetric error)
EST. DFCT	=	$1/2\ \varepsilon_5^2$
DFCT (±)	=	$F(\underline{a}^*) - F_i\ (\underline{a}, \pm \varepsilon_5 \cdot \sigma_i)$
ERROR (±)	=	positive and negative error S_i, resp.
NFCT	=	no. of iterations
X	=	no. of fixed parameters

Result print-out:

FCT	=	$F(\underline{a}^*)$
NFCT	=	no. of iterations
F	=	flag NF
I	=	index of parameter
PA (I)	=	parameter a_i^*
SYM. SD	=	σ_i (one st. dev.) = $(E_{ii})^{1/2}$
CORR	=	correlation matrix $\varrho_{ik} = E_{ik}/(E_{ii} \cdot E_{kk})^{1/2}$

Table 4: Explanation of print-out

Figure Captions

Fig. 1 Curve $F(\underline{a}) = \text{const}$ in parameter space for the case of two parameters and the quadratic approximation using the first and second derivatives of $F(\underline{a})$ at $\underline{a} = \underline{a}_o$. The steps $\underline{\Delta a}_1$ and $\underline{\Delta a}_2$ are the results of the two methods explained in the text.

Fig. 2 Geometrical meaning of the calculation of the asymmetric error s_j according to eq. (20) explained in the text.

Fig.1

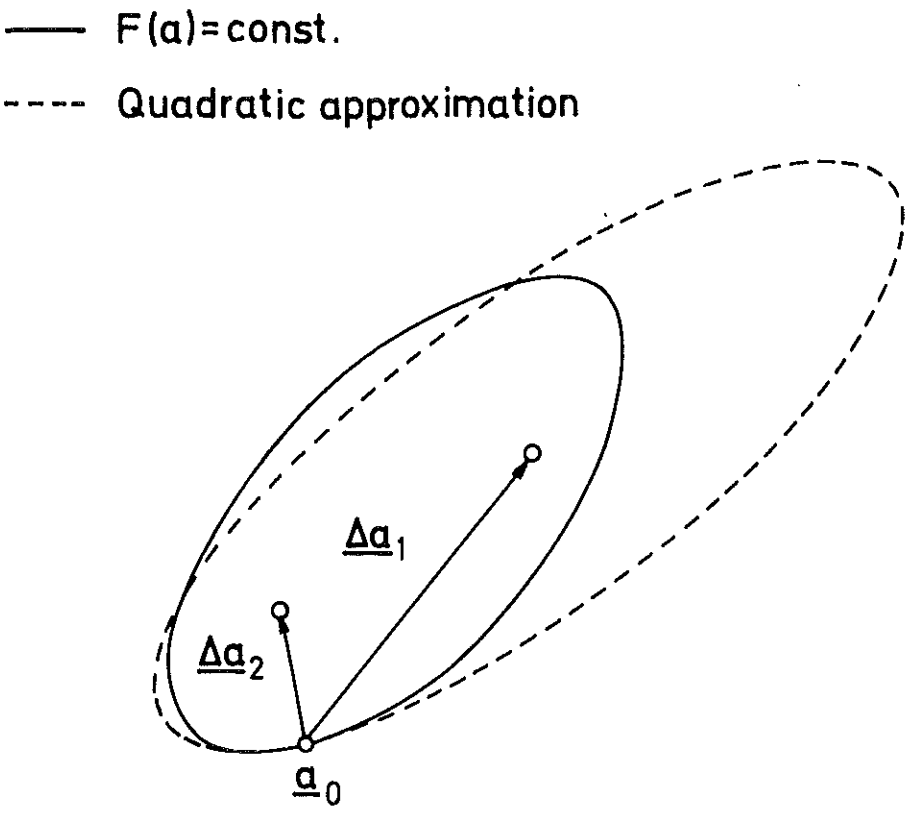
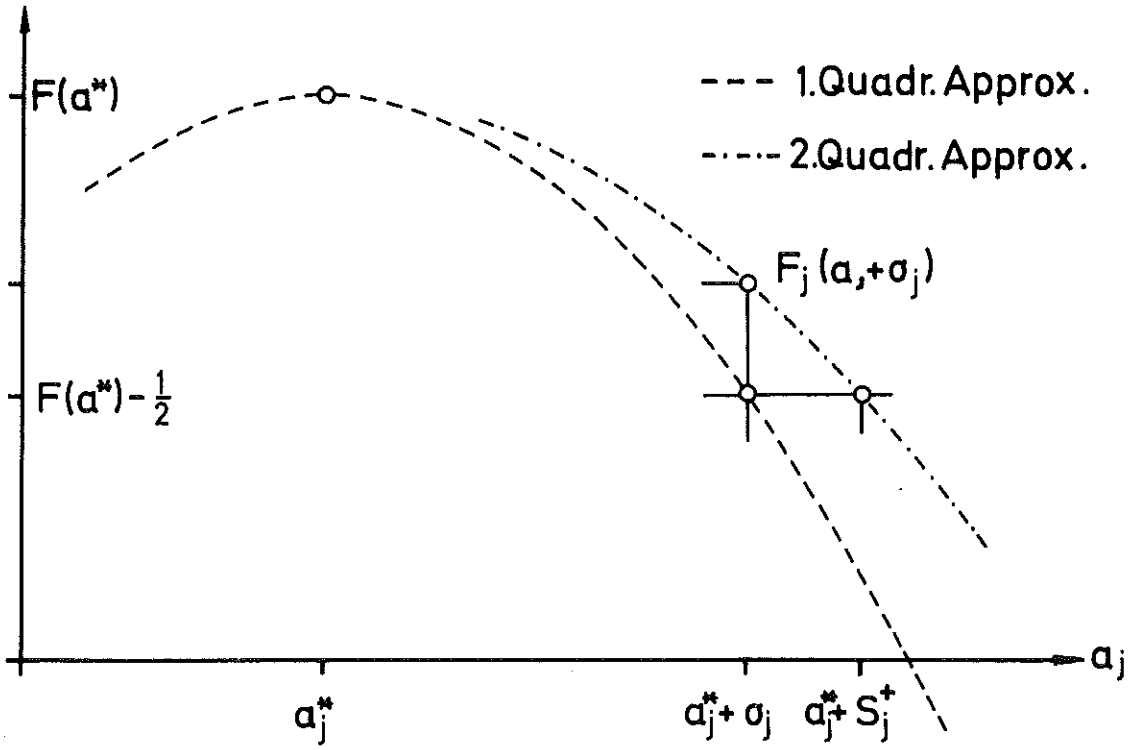

F(a)=const.
Quadratic approximation
Δa1
Δa2
a0

Fig.2


F(a*)
F(a*)-½
1.Quadr. Approx.
2.Quadr. Approx.
Fj (a,+σj)
aj
aj*
aj*+ σj
aj*+ Sj+