User's Guide

The language is F-90. No syntax difference with F-77 apart from the arbitrary length of strings. No line breaks in this version. The intrinsic function CPU_TIME (F-95) is in use. The calls of it may be converted to comment in absence of this function in the compiler.

1 Program call

It is assumed below that all non-integer values, apart from `EXDA`, are `REAL*8`.

The call has the following form:

```
HSQ=FUMILIM(A,NAM,NP,EXDA,LEXD,NEXD,US,NIT,OUT,OP,NDIM).
```

Here

- `HSQ`=χ^2`/NDF` (Number of Degrees of Freedom, `NDF=NEXD-NP+1`);
- `NP` – number of parameters (not restricted);
- `A(NP,6)` – parameters' array;
- `NAM` – the name of the file with the parameter initial values or `'*'` – no parameter names, the initial values are defined by the user in `A(NP,6)`.
  If there is the list of names in user disposal, he can create the file with the parameter initial values, using

  ```
  CALL FUMINI('FILENAME',HNAMES,A,NP),
  ```

  where `HNAMES` - array of names ($\leq$15 symbols).
- `EXDA(LEXD,NEXD)` – experimental points array, Here `LEXD` is the number of units belonging to one experimental point, `NEXD` is the number of experimental points. To avoid computer memory problems, when number of experimental points is huge, the corresponding array (if any) must be declared as **REAL**;

- US – user's subroutine with arbitrary name(must be declared as EXTERNAL);
- NIT – is the iteration limit;
- OUT – file name to print (OUT='*' – screen output, OUT='-' – no output), any other set of symbols is the output file name; when (OUT' is the file name, short line of information about each iteration is printed to screen.
- OP – (options) character string to control the input/output, OP='*' means the default input/output;
- NDIM – the dimension of experimental points and output parameters of US.

Parameters array has the following structure:

- A(J,1) – initial/final value of parameter [A];
- A(J,2) – parameter error (final)[SIGMA];
- A(J,3) – saved initial value of the parameter;
- A(J,4) – the initial parameter step restriction [PL], A(J,4)<0 means the parameter fixed by the user, A(J,4)=0 (not defined) means that this value should be adjusted by FUMILIM (recommended);
- A(J,5:6) – lower and upper bounds on the parameter value in any order. When A(J,5)=A(J,6) (in particular when not defined) the bounds are $\pm\infty$.

If the file with the name NAM exists (any name, [any extension]), the initial data will be taken from this file. The file format is as follows:

```
1 NAME(1) A(1,1) A(1,4) A(1,5) A(1,6)
...
...
NP NAME(NP) A(NP,1) A(NP,4) A(NP,5) A(NP,6)
```

For tasks, linear respectively parameters, it is permissible to have:

```
I NAME(I) 4*0.
```

Parameter number is defined by the string number in the initial data file. The numbers in the first column are ignored and used only for better orientation of the user in his file.

Names in this file must not contain intrinsic commas or blanks.

`US` must have four parameters: output value(s), argument(s), parameters, derivatives, respectively. An arbitrary number of derivatives can be defined analytically.

In case of `NDIM==1` (scalar case) the simplest user function without analytical derivatives can look like below:

```
    SUBROUTINE SIMPL(Y,X,C,DF)
IMPLICIT REAL*8(A-H,O-Z)
DIMENSION A(*)
 Y=A(1)+A(2)*X
    END
```

For the 3-dimension case (space) and 9-elements experimental points with the default structure the user's subroutine can looks like:

```
    SUBROUTINE SIMPL3D(Y,X,C,DF)
IMPLICIT REAL*8(A-H,O-Z)
DIMENSION X(*),Y(*),A(*)
 x=X(1) ! experimental arguments
 y=X(2)
 z=X(3)
Y(1)=A(1)*x**2 + A(2)*y*z
Y(2)=A(3)*y**2 + A(4)*x*z
Y(3)=A(5)*z**2 + A(6)*x*y
    END
```

For "u"-option and 6-elements experimental points we will have

```
    SUBROUTINE SIMPL3U(Y,X,C,DF)
IMPLICIT REAL*8(A-H,O-Z)
DIMENSION X(*),Y(*),A(*)
DATA Err/0.01/
 Yx=X(1) ! experimental values
 Yy=X(2)
 Yz=X(3)
 Xx=X(4) ! experimental arguments
 Xy=X(5)
 Xz=X(6)
THEORx=A(1)*Xx**2 + A(2)*Xy*Xz
THEORy=A(3)*Xy**2 + A(4)*Xx*Xz
THEORz=A(5)*Xz**2 + A(6)*Xx*Xy
 Y(1)=(Yx - THEORx)/Err
 Y(2)=(Yy - THEORy)/Err
 Y(3)=(Yz - THEORz)/Err
    END
```

Some output values after the fit can be found in

```
COMMON/FUMOUTPUT/HISQ,AKAP,DT,IEND,IITER,NWRONG
```

Here HISQ=$\sum \chi^2$, AKAP is the Estimated Distance to Minimum, DT is Elapsed Time in seconds, IITER is the last iteration number, NWRONG is the number of the ignored experimental points (if any). The values of IEND are as follows:

```
1 -- Full Success
2 -- Success after Fixing Some Parameters
3 -- Fit Terminated, No Further Convergence
4 -- Fit Terminated, Iteration Limit
5 -- Fit Terminated, No Free Parameters
6 -- Wrong Input Data
```

In the latter case the fit terminated with the function value of 1d+30 and diagnostic string with the first two symbols: **.

After finish of each iteration information about current value of χ^2, Kapp and Lamb printed. Here Kapp the Estimated Distance to Minimum. The fit is finished when Kapp<0.01. Lamb is the minimum ratio of parameter step restriction A(i,4) to suggested by FUMILIM parameter increment. When Lamb after 0-iteration is too small, the user should think about increasing of A(i,4), because such a situation leads to increasing of number of iterations.

In some tasks the minimization process jumps over the minimum. In this case the program retreats to the parameter values of the previous iteration and recalculates the parameter increments using the decreased parameter steps to find the first derivatives.

The simplest call of FUMILIM looks like

```
HSQ=FUMILIM(A,'*',NP,EXDA,LEXD,NEXD,US,NIT,'*','*',1).
```

The first star in this call means no initial data file, the second one means the print to screen. The third star means default output, i.e. print full information about the 0-th and the last iterations (only one string of information for intermediate iterations).

The call of add-on program to take experimental points from the file is shown below.

```
HSQ=FUMIFILES(NAM,NP,HEXD,LEXD,NEXD,US,NIT,OUT,OP,NDIM).
```

Here HEXD - file name (any name and extension) with experimental points. The other parameters are the same as in main call. No any arrays should be declared in this case in the calling program. When the user does not remember exactly the number NEXD, he/she can use the certainly more value.

In this case the program will work with the real number of experimental points, contained in the file. When the experimental data array is formatted so that all elements of experimental point are ordered in one line, then any comment after the last element of the experimental point is permissible. On the author's opinion, this call is preferable when the user works with experimental data for publishing. When the fit involved into the events reconstruction, of course, the direct call of `FUMILIM` is necessary.

2 Parallel fit

The **openMP** service is used to provide the parallel fit. The factor of acceleration for scan-procedure can achieve the number of parallel processors. For the standard fit paralleling is applied when calculating numerical derivatives on parameters. So, factor of acceleration can be close to number of active numerical derivatives for the current experimental point. For light `US`, like polynomial, this option is not recommended.

It is assumed that commands, initializing this process, namely

```
!$ include 'omp_lib.h'
!$ nproc=omp_get_num_procs()
!$ call omp_set_num_threads(nproc)
```

is placed in the first lines of main program. The corresponding directive (different for different compilers) should be placed in command line. The `US` must not contain variables, depending on parameters, in common-blocks.

3 Input/Output control

The input/output are controlled by the character string `OP` (options).

3.1 Input control

- `OP`$\bigcup$`'u'` – user defined structure of the experimental point.
- `OP`$\bigcup$`'w'` – the fit with ignoring wrong points (`IWP`) with large χ^2.
- `OP`$\bigcup$`'e'` – the fit with the error correction is ordered (for the data, subdivided in sets).
- `OP`$\bigcup$`'s'` – the procedure of preliminary scan is ordered.
- `OP`$\bigcup$`'t'` – transformation of exp.p. array.
- `OP`$\bigcup$`'l'` – logarithm of exp.p. array.
- `OP`$\bigcup$`'f'` – fixing of parameters.

Any combination and order of the symbols mentioned above is possible in the string `OP`.

option "u"
Traditionally, the default format of the experimental point is the following:

$$Y_1, Err_1[, Y_2, Err_2...], x[x_2, ..],$$

where Y_i – experimental value, Err_i – experimental error, $x...$ – arguments. $US(y, x, a, df)$ receives the array of arguments (x) to form theor. value y. Then `FUMILIM` forms the value

$$\chi_i = \frac{Y_i - y_i}{Err_i}. \tag{1}$$

Here $\Sigma\chi^2$ is the subject of minimization.

Unfortunately, not all the tasks can be solved within this approach. To rectify the situation, the option "u" can be used. In this case the x-array in the `US` is the whole experimental point. The user can do with it, what is necessary on his opinion, but the result value y must be in the form of Eq.1.

Here we consider the fragment of the real task when this option is capable to help. In the investigation of polarization phenomena the asymmetry is defined as follows:

$$R = \frac{N_+ - N_-}{N_+ + N_-}, \tag{2}$$

where indexes denote opposite states of polarization, $N_{+/-}$ – numbers of events. So, we can construct an experimental point having two experimental values and one argument (for example, scattering angle). To find the beam polarization, when the analyzing power is known, the following `US` was used:

```
            SUBROUTINE USER(Y,X,A,DF)
   IMPLICIT REAL*8(A-H,O-Z)
   DIMENSION X(*),A(*),DF(*)
!   X(1): EXP.VALUE N-
!   X(2): EXP.VALUE N+
!   X(3): SCATTERING ANGLE
      E1=X(1)
      E2=X(2)
      DISPER=(1/E1+1/E2)
      ERR=SQRT(DISPER)
      F=T20(X(3))  ! ANALYZING POWER
      R=(A(1)*E1 - E2)/(A(1)*E1 + E2)
      Y=(R - A(2)*F)/ERR
      DF(2)=-F/ERR
            END
```

Here `A(1)` is the parameter taking into account a possible different beam intensity in a different polarization mode or something like this, `A(2)` is searched value of the beam polarization. Of course, the task can be solved with the standard format of the experimental point. The value of `R`, as the experimental value can be used. But in this case, the problem of the coefficient `A(1)` should be solved beyond the fitting program.

option "w"

This option allows one to ignore experimental points with a huge contribution to $\sum \chi^2$ (see Eq.1).

Firstly, the fit is fulfilled without any restrictions. The second fit ignores the points with

$$\chi_i^2 > hifac\langle\chi_0^2\rangle, \tag{3}$$

where

$$\langle\chi_0^2\rangle = \frac{\sum \chi^2}{nexd}$$

obtained in the previous sub-fit. If new obtained value of $\langle\chi_1^2\rangle$ is essentially less than $\langle\chi_0^2\rangle$ the new sub-fit with new restriction will be continued. This sequens of sub-fits will be finished when

$$\langle\chi_i^2\rangle > 0.95\langle\chi_{i-1}^2\rangle$$

The default value of $hifac$ in Eq.3 is 30. It can be redefined by the user in

`COMMON/FUMHISQ/hifac,iforbid`.

If experimental data are subdivided in sets, for each set $\langle\chi_i^2\rangle$ is calculated separately (**new**).

Sometimes it is necessary to forbid deleting of experimental points with the bound values of arguments. To realize this option, the user should define `iforbid=N` (see common-block above), where `N` is the position of such an argument (only one) in the array of the experimental point (**new**).

In Fig.1,2 it is shown applying of this method in the real life. All tables for these pictures are prepared by `FUMILIM`.

option "e"

This option can be applied when the experimental points are subdivided in sets. Frequently, errors of measurement are not known well enough, but when the number of experimental points is rather high, $\langle\chi^2\rangle$ is a good indicator of the measurements' accuracy. The weigh of each set is defined as

$$w(j) = \frac{1}{w_{max} \cdot \langle\chi_j^2\rangle},$$

where $w_{max} = 1/\langle\chi_j^2\rangle_{min}$. One can see that the set with the least $\langle\chi_j^2\rangle$ is taken into account with $w_j = 1$.

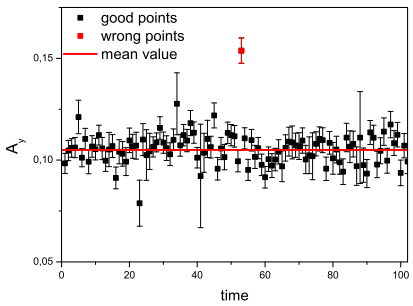

Figure 1: Measurement of beam polarization.

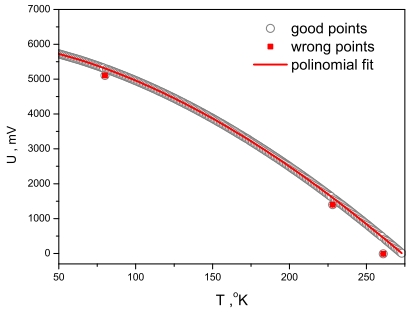

Figure 2: Calibration of the thermocouple.

When options "w" and "e" are used together, the fit is fulfilled in 3 steps. The first one: no "w", no "e" is applied. The second one: only "w" is applied. The third one: both "w" and "e" are applied.

option "s"

Functions with a plurality of local minima are a difficult task for `FUMILIM`. Likely, the nearest local minimum will be found. The scan-procedure helps to find the preliminary global minimum. Then the standard fit follows.

All scanned parameters must have limits for parameter values. Two modes of choice of parameters to scan are available. The first one when parameters for scanning defined directly by the user in

```
COMMON/NUMSCA/NUMSCAN(19).
```

If $numscan(1) == 0$, the second mode is fulfilled. In this case all parameters with defined limits are considered as candidates. At the central parameter values between limits the gradients

$$G(i) = \frac{\partial \sum \chi^2}{\partial A(i)}$$

are calculated. The parameter is accepted to scan when

$$\frac{G(i)}{maxval(G)} > Gr.$$

Default value of G is 0.05. It can be redefined in

```
common/fumsca/Gr,npsc,nexsc,nitsc
```

Here `npsc` is the number of scanned values, by default `npsc`=3. `nexsc` is the restriction for the number of experimental points per set, taken for scanning. By default, `nexsc`=100. If the number of experimental points is above the restriction, the selected points are distributed uniformly in the array of experimental points. `nitsc` is a number of iteration. By default, `nitsc`=20. After each iteration the range of scanning for each parameter is decreased by 30%. The exit from the cycle take place when $\chi^2_i > 0.9\chi^2_{i-1}$.

The number of combinations for the scanned parameters is

$$(NPSC)^{NP},$$

where NP is the number of scanned parameters. The order of parameters to scan is descending relatively the gradient values. As it is seen from the above relation, the excessive number of parameters to scan leads to huge time-consuming.

option "t"

Rather frequently exp.point array has the structure

$$x, y, err.$$

`FUMILIM` can work with these data when the option "t" is ON.

option "l"

A lot of spectra in physics can be described by

$$F(x, A) = exp(f(x, A)), \tag{4}$$

where $f(x, A)$ is linear or almost linear function respectively parameters A. In such cases it is recommended to use

$$log(y), err/y, x$$

instead

$$y, err, x$$

as experimental data and $f(x, A)$ instead $F(x, A)$ as `US`. Such a fit is faster, more reliable and allows one instead of scrupulous definition of parameter initial values not to define them at all. The option "l" (logarithm) makes the corresponding converting of the exp.p. array.

3.2 Output control

Options '+', '-' are active always. The other options are active when `OUT` is the file name (**new**). All ordered output files has the name `'H.XY'` where `'H'` is `OUT` before '.' (if any) (**new**).

- `OP`$\bigcup$`'-'` – The information accompanying the program self-tuning does not print.
- `OP`$\bigcup$`'+'` – full information about each iteration goes to the file `OUT`. By default, full information is provided only for the 0-th and last iterations.
- `OP`$\bigcup$`'o'` – the final parameter values are recorded to 'H.fp' in the format of the initial data file.
- `OP`$\bigcup$`'O'` – the final parameter values are recorded to 'H.fe' in the format: line number, parameter name, parameter value, parameter value error.
- `OP`$\bigcup$`'h'` (histogram) – the 100-channel histogram of $\lg(\chi^2/\chi^2_0)$ is made in the file `H.hi`. The string format is as follows: $x, N1, N2$, where $N1$ is the number of events below the χ^2-constraint, $N2$ – above.
- `OP`$\bigcup$`'p'` (plot) – the individual χ^2-histograms for each set are made in the file `H.pl`. When there is at least one wrong point, we have two columns per set, defined as in the previous statement, otherwise 1 column per set.
- `OP`$\bigcup$`'c'`(curve) – the 100-channel table of the function, describing the data, is built in the file `H.cu`. Its format is as follows: $x, y, y-er, y+er$, where $y \pm er$ - confidence level. **This option is available only for standard structure of exp.points, described by function of one argument.** For multi-argument function one can use `H.ap`, choosing appropriate argument as x and column 'Y_theor' as y. For the user defined structure of the experimental points to estimate the deviation of the description from experimental points the 'dY/sig' column as y is good.
- `OP`$\bigcup$`'C'` – with this option exp.poits together with describing curve is built in the file `H.et` (experiment-theory). When some intrinsic functions (see below) is used, the components are recorded too. The format is:

```
x, Y_exp, err, Y_theor, [components]
```

- OP⋃'a'(all) – the list of all points is recorded to the file H.ap. The structure of data is as follows: $Y_{theor}, \sigma_{theor}, (Y_{exp} - Y_{theor})/\sigma_{exp}, \chi^2$, set number, dimension number. Then the data of the experimental point follow. The wrong points are labeled in the last column by "W" (if any).

- OP⋃'W'(Wrong) – the list of wrong points is recorded to the file H.wp. The structure of data is the same as in the previous item,'W' absorbs 'w'.

- OP⋃'z' – the complete table of the parameter pair correlations is recorded to the file H.zm. The column of gradients is also presented in this file.

Each record of files is accompanied by the message line in the OUT file.

4 Intrinsic user's subroutines

A number of popular user's subroutines is included in the package.

4.1 polynomial fit

$$y = \sum_i^n A(i) \cdot x^{i-1} \tag{5}$$

$$y = \sum_i^n A(i) \cdot x^i \tag{6}$$

where $A(i)$ are parameters, n is number of parameters. Any parameter initial values are acceptable. The name of subroutine,connected to Eq.5, is FUMPOL, the other name is FUMPOL0.

4.2 functions with exponent

$$y = c_1 \cdot \exp(c_2 \cdot x), \quad \textit{exponent distribution},$$

$$y = c_1 \cdot x \cdot \exp(c_2 \cdot x^2), \quad \textit{Rayleigh distribution},$$
$$y = c_1 \cdot \exp(c_2 \cdot (x - c_3)^2), \quad \textit{Gauss distribution},$$

where c_1, c_2, c_3 – parameters. It is assumed that c_2 is everywhere negative. The data can be described by one or sum of several functions. The number of integrable functions is `NP/2` for the exponent and Rayleigh function and `NP/3` for Gauss one, where `NP` is the number of parameters in call of `FUMILIM`.

The fit will be successful when parameter initial values are rather close to right ones. The subroutine names are `FUMEXP`, `FUMRAY` and `FUMGAU`, respectively. The `FUMEXP` is good for description of scattering spectra:

$$\frac{d\sigma}{dt} = \sum_i a_i exp(-b_i t),$$

where t is the Mandelstam variable.

The distribution like Gauss + uniform background is also can be fitted by `FUMGAU`, because

$$c_1 \cdot \exp(c_2 \cdot x^2) == c1,$$

when $c_2 = 0$.

For the experimental points, ordered ascending argument, it is suggested by the package the intrinsic adjustment of the parameter initial values before the standard fit. So, the fit for Rayleigh and Gauss one- and two-peak distributions can be successful without definition of any doublet/triplet of parameters (defined as 0). The first and last doublet/triplet of parameters are found especially close to the final parameter values. If the parameters are defined in part, they should be in descending order respectively the slope parameters.

Each doublet/triplet initial values can be defined or not by the user. The sign of definition is the nonzero normalization coefficient.

The fit by these function of real data is included into the package of the testing program.

5 Add-on programs

5.1 Work with histograms

We mean that histogram is two columns data (x,y).

In call

```
HSQ=FUMHIST(NAM,NP,HEXD,NEXD,ERR,SUB,NIT,OUT,OP)
```

`HEXD` is name of file with experimental points, positive value of `ERR` means common value of experimental error for all points. Otherwise experimental error is $\sqrt{y}$. Other parameters in this call have the same sense, as in main call. The value of `NEXD` can exceed the real value of strings in `HEXD`.

5.2 Multi-polynomial fit of data

The program is predestinated to describe data by multi-polynomial, each for his own range of x.

```
HSQ=POLIMULT(NPOL,NSET,HEXD,NEXD,FUN,NIT,OUT,OP)
```

Here
`NPOL` – number of terms of the polynomial,
`NSET` – number of regions of x,
`HEXD` – name of file with data,
`NEXD` – number of experimental points (can exceed real number of experimental points).
The structure of data file is
y, y_{err}, x, or
x, y, y_{err} (option in this case `'t'` should be incorporated into `OP`).

The number of a polynomial, i, belonging to a certain domain of x is defined by the following equations:

$$\begin{aligned} step &= \frac{x_{max} - x_{min}}{nset} \\ i &= \frac{x - x_{min}}{step} + 1. \end{aligned} \tag{7}$$

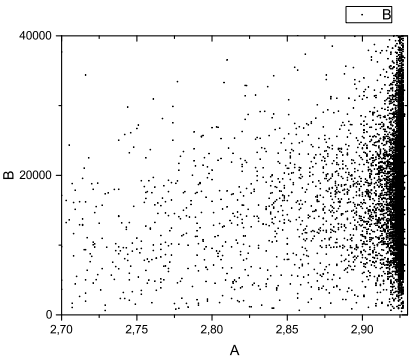

Figure 3: Input data.

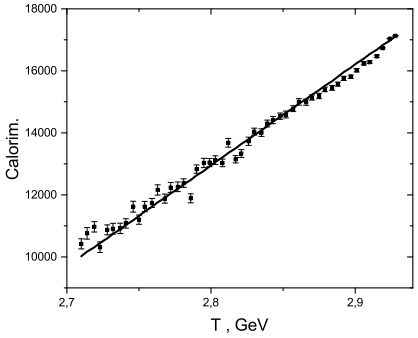

Figure 4: Output data.

5.3 xy-regression

Add-on program, using `FUMILIM`, allows one to fit the statistical dependence y on x.

The call has the following form:

```
HSQ=FUMREGRES(NAM,NP,US,FILNAM,NEXD,XMIN,XMAX,YMIN,YMAX,FAC).
```

The parameter names, having the same denotation as in main call at the start of the user's guide, have the same sense.
`FILNAM` - file name with data, having format
x1, y1
x2, y2...
and so forth.
The value of `NEXD` can exceed real number of xy-pairs.
`XMIN -- YMAX` are limits for xy-values, involved in the fit.
`US` should be declared as `EXTERNAL`.

The first step of the program work is creating of two-dimensional plot 50x50 for xy-values. The plot parameters is formed proceeding from the condition
`XMIN=MAX(XMIN,xmin)`, where `xmin` is minimum value of x found in the input array. Analogically the other limits are formed. So, if the user want to use all experimental points from the input array, he/she can define the corresponding parameter values as the huge negative and positive values.

The next step is search of mean value and its error of 1-dimensional y-distribution for given x-slice. The tails of the distribution with values less

then FAC*M are rejected. Here M is the maximum value of the y-distribution. The third step is the fit of data with the suggested by the user function. The fit results are in 'regres.res', the obtained by the described above way experimental points and the fitting curve are saved in the file 'regres.et' (see option "C" above). One can see results in Figs.3,4. This program was used for investigation of dependence of the calorimeter signal on proton kinetic energy. The fit was fulfilled for $2 \cdot 10^6$ xy-pairs.

5.4 Analyzing power on polarized beams

The program calculates analyzing power of the reaction $A_y(\theta)$ for vector polarized beam or T_{20}, T_{22} for tensor one from φ/θ-plots, obtained on polarized beams. When calculating analyzing power, it is assumed that two of three $\cos\varphi/\theta$-plots for different modes, obtained during off-line analysis, are used.

The call has the following form:

```
CALL MAKEANPW(IAR1,IAR2,CMI,CMA,NC,TMI,TMA,NT,P,C21,IAN)
```

Here IAR1(NC,NT),IAR2(NC,NT) are φ/θ-plots, CMI,CMA,NC,TMI,TMA,NT are limits and number of bins for φ/θ-plots, P(1), P(2) – beam polarization values in cartesian coordinates for IAR1,IAR2, respectively. C21 is the ratio of intensities in polarization modes 2 and 1. It is implicit vector polarization for IAN=1 and tensor one for IAN=2. Other variables, instead of θ, can be used in the user's task. $\cos\varphi$ bins are implicit for vector polarization and $\cos 2\varphi$ for tensor one. So, for 2π geometry CMI=-1,CMA=1.

Values of vector analyzing power a_j are searched by fit

$$\sum_{i,j}(R_{ij} - a_j x_i - a_0)^2 = min. \tag{8}$$

Here x_i is values of $\cos\varphi$ for i-th slice in plots,

$$R_{ij} = \frac{C \cdot N_{ij}(1) - N_{ij}(2)}{C \cdot N_{ij}(1) + N_{ij}(2)} \cdot \frac{2}{(P(1) - P(2))}, \tag{9}$$

$$\Delta R_{ij} = \sqrt{\frac{N_{ij}(1)N_{ij}(2)}{[N_{ij}(1) + N_{ij}(2)]^3}} \cdot \frac{2}{(P(1) - P(2))} \tag{10}$$

where $N_{ij}(k)$ are bin volumes of plots, $P(k)$ is the beam polarization, $k = 1, 2$ relates to the polarization mode. a_0 takes into account difference of intensities

in different polarization modes. In Eq.9 $C = 1$ in this case. Because of low pair correlation of a_0 with a_j (< 0.02), such an approach can be used instead of scrupulous (and sometimes wrong) evaluation of intensities in different polarization modes.

Values of tensor analyzing power a_{0j}, a_{1j} are searched by fit

$$\sum_{i,j}(R_{ij} - a_{1j}x_i - a_{0j})^2 = min, \tag{11}$$

where x is $cos2\varphi$. In Eq.9 $C = C21$ in this case.

The result of the fit of vector analyzing power Ay is written to file 'Ay.dat'.

Results of the fit of tensor analyzing power are written to files 'T20T22.dat' for $T20, T22$ values and to 'A0A1.dat' for a_{0j}, a_{1j} values. $T20, T22$ relates to a_{0j}, a_{1j} as follows:

$$T_{20} = -\frac{1}{2\sqrt{2}}a_{0j} \tag{12}$$

$$T_{22}(j) = -\frac{\sqrt{3}}{2}a_{1j} \tag{13}$$

Fit details is written to 'AnPw.res'.

5.5 Event Reconstruction

The call of `FUMILIM` which looks like

```
HSQ=FUMILIM(A,'ANY',NP,EXDA,LEXD,NEXD,US,NIT,'-',OP,1).
```

means no output at all except the initial values in the parameter array will be replaced by the final ones. The value of the second program parameter in this call is ignored. Such a call is useful when `FUMILIM` is involved into the event reconstruction and the multiple call is assumed. The speed optimization is envisaged with this option starting from the second call, if the following conditions are provided:

- no fixed parameters;
- no bounds for the parameter values;
- NDIM=1, 1-dimensional exp. points.

The program recognizes the subsequent calls due to special code, recorded into the initial data array during the first call, when the above conditions are fulfilled. The fast branches of the package are envisaged for this case. Especially the fast branch is realized for two parameters (straight track line).

For the reconstruction of the straight track line via drift chambers or straws, which can be combined with single detectors, the add-on program is created:

```
HSQ=FUMDRIFT(OUTP,EXD,ERR,NP,NDRIFT,NSINGLE,AMIN,AMAX),
```

where `OUTP` is the output parameter array, `EXD` is the array of experimental points, `ERR` is the error of the distance measurement, `NP` is number of parameters, which can be either 2 or 3. First two parameters are angle and profile track coefficients. Usually `NP=2`. In cases when start time-signal has the sizeable divergence, it is helpful to use `NP=3`. Then program searches for common additive correction for all distance-values of experimental points. At `NP=3` program works approximately two times slower in comparison with `NP=2`. `Ndrift` is the number of drift chamber planes, `Nsingle(>=0)` is the number of single detectors, `Amin,Amax` are restrictions for the angle coefficients. (A_{min}, A_{max}) is the range of acceptable solutions for the angle coefficient. As a rule, it is unchangeable for the whole experiment. These restrictions allow the user to avoid wrong solutions in the straw case.

Declaration for `OUTP` is
`DIMENSION OUTP(NP)`,
declaration for `EXDA` is
REAL EXDA(3,NDRIFT+NSINGLE).
Structure of the experimental point for drift chambers is:
`EXDA(1,I)` – distance after time-space transforming;
`EXDA(2,I)` – x(y)-coordinate of the drift chamber wire;
`EXDA(3,I)` – z-coordinate of the drift chamber plane.
Structure of the experimental point for single detectors is:
`EXDA(1,I)` – x(y) coordinate;
`EXDA(2,I)` – error of x(y)-coordinate;
`EXDA(3,I)` – z-coordinate of the detector plane.
After this call `HSQ` is the χ^2/NDF, in `OUTP(1:2)` are found values of angle and profile parameters, respectively. `OUT(3)` (if any) is the additive corrections for all distance values. Ascending/descending order of z-coordinates in the `EXDA`-array for drift chambers is desired. It is assumed that experimental points for single detectors follow after ones for drift cambers.

6 Tricks

The variable `IEXP` in

```
common/fumdebug/IEXP
```

shows the experimental point number. This value can be used for bifurcations. Also, it can be used to accelerate the program. The total number of `US` calls for each experimental point is equal to $n+1$, where n is the number of parameters. The first call is with current values of parameters. Then n calls follows, when each of parameters varies to find derivatives (if they are not defined analytically). Each of these calls take place at the same value of `IEXP`. So the part of calculations, not depending on parameters, is repeated many times with the same result.

Below is the example how to avoid these repetitions.

```
      SUBROUTINE POLINOME(Y,X,A,DF)
     COMMON/FUMDEBUG/IEXP
     COMMON/XPW/XPW1,XPW2
     DIMENSION A(NP)
      data lastexp/0/

     IF(iexp/=lastexp) THEN
       lastexp=iexp
      XPW1=SQRT(1+X**2)
      XPW2=SQRT(1+X**3)
     ENDIF
!    SAVE   is possible for single processor fit
!           instead of COMMON/XPW/XPW1,XPW2
     Y=A(1)*XPW1+A(2)*XPW2
        END
```

Pay attention that values, calculated during the first entry, should be placed in common-blocks.

6.1 Self-tuning

The main achievement of this package is possibility to work properly with tasks having the huge number of parameters. In this case it is assumed that experimental data are subdivided in sets, each of them is described by by reasonable number of parameters. So, the task with turns of satellite "Messenger" around Mercury has 4000 turns. Each turn is described by 6 global and 7 local parameters, individual for each turn. So, the total number of parameters for 100 turns is 706, while the number of parameters, described the separate set, is 13, easy solvable task. In principle, such a task is solvable without self-tuning. But in this case during each iteration for each experimental point 706 calls to the user subroutine will be done to find $\partial F/\partial A(i)$, where i is the parameter number. (706-13) calls will produce $\partial F/\partial A(i) = 0$, but time consuming for these calls is the same as for effective ones, and that is most time consuming part for the task. During self-tuning the table is formed which contains the list of effective derivatives for each set. And in our example we should take only 13 calls for each experimental point. Acceleration is $706/13 \simeq 54$ times.

Now our task to accelerate self-tuning subroutine itself which still should take all 706 call for each experimental point to prepare the necessary table. We do this in the following way. To the array of each experimental point we incorporate new value, the turn number. Then we inform `FUMILIM` about position of this value in the array of experimental point via

`common/fumset/kset`

The value of number, responsible for belonging of the experimental point to the certain set, can be non-integer and negative.

When $kset \neq 0$, only one experimental point from the set is investigated for the presence of derivatives. In our example each turn contains about 1000 experimental points. So, we accelerated the self-tuning subroutine 1000 times.

In the previous version of `FUMILIM` two experimental point, having the same kit of numerical derivatives, belonged to the same set. That solution in principle is right, because the table of parameters to calculate for each set is created only for numerical derivatives. But sometime it is uncomfortable. So, two sets with the same kit of numerical derivatives, but having the different analytical derivative for the normalization factor, was treated as one set, when the user would like to know χ^2 separately for each of these sets. To overcome this circumstance two options, 'g' and 'G', were introduced into the package, but it was not the best solution. Now all derivatives for the two experimental point must be identical to belong the same set. So, mentioned above options are obsolete.

6.2 Creating of files of initial parameter values for multi-set tasks

When the number of parameters in the problem is hundreds, the creation of an initial data file becomes tedious. The next subprogram will help to facilitate this process.

```
CALL FUMMULT(HI,HF,NGLO,NLOC,NSET,IST)
```

Here `NP` is the number of parameters for `FUMILIM`, which is `NP=NGLO+NLOC*NSET`, `HI` is the name of initial file, `HF` is name of final file, `NGLO` is the number of global parameters, `NLOC` is the number local parameters, `NSET` is the number of sets, `IST` start number for the set numeration. Here it is assumed that all sets have the same number of local parameters with same start values. Below is example, where the initial file have NGLO=6, NLOC=7

```
1   d0     -2.0090E+02   0  -2.1500E+02  -1.7500E+02
2   dx0    -1.7656E-03   0  -1.0000E-01   2.0000E-01
3   dy0    -4.2147E-03   0  -1.0000E-01   2.0000E-01
4   dz0     1.7859E-01   0  -3.0000E-01   3.0000E-01
5   lat     2.0048E+00   0  -2.0000E+01   2.0000E+01
6   long    1.9944E+02   0  -3.4000E+02   6.0000E+02
7   t      8.3838E+01   9.2385E-01   2.1000E+01   2.8000E+02
8   R1     1.6309E+00   1.7758E-02   0.1000E+00   2.4000E+00
```

```
 9   R2     1.1504E+00   1.2925E-02   0.1000E+00   2.6000E+00
10   Z0     -3.0532E-01   4.3194E-03  -15.0000E+00   15.0000E-00
11   bimfx  0.0000E+00   3.5000E-01  -19.0000E+00   19.0000E+00
12   bimfy  0.0000E+00   1.4000E+00  -19.0000E+00   19.0000E+00
13   bimfz  0.0000E+00   1.4000E+00  -19.0000E+00   29.0000E+00
```

After transformation it to the final file with `NSET=2, IST=28` we have

```
 1   d0              -2.0090E+02   0.0000E+00  -2.1500E+02  -1.7500E+02
 2   dx0             -1.7656E-03   0.0000E+00  -1.0000E-01   2.0000E-01
 3   dy0             -4.2147E-03   0.0000E+00  -1.0000E-01   2.0000E-01
 4   dz0              1.7859E-01   0.0000E+00  -3.0000E-01   3.0000E-01
 5   lat              2.0048E+00   0.0000E+00  -2.0000E+01   2.0000E+01
 6   long             1.9944E+02   0.0000E+00  -3.4000E+02   6.0000E+02
 7   t_________0028   8.3838E+01   9.2385E-01   2.1000E+01   2.8000E+02
 8   R1________0028   1.6309E+00   1.7758E-02   1.0000E-01   2.4000E+00
 9   R2________0028   1.1504E+00   1.2925E-02   1.0000E-01   2.6000E+00
10   Z0________0028  -3.0532E-01   4.3194E-03  -1.5000E+01   1.5000E+01
11   bimfx_____0028   0.0000E+00   3.5000E-01  -1.9000E+01   1.9000E+01
12   bimfy_____0028   0.0000E+00   1.4000E+00  -1.9000E+01   1.9000E+01
13   bimfz_____0028   0.0000E+00   1.4000E+00  -1.9000E+01   2.9000E+01
14   t_________0029   8.3838E+01   9.2385E-01   2.1000E+01   2.8000E+02
15   R1________0029   1.6309E+00   1.7758E-02   1.0000E-01   2.4000E+00
16   R2________0029   1.1504E+00   1.2925E-02   1.0000E-01   2.6000E+00
17   Z0________0029  -3.0532E-01   4.3194E-03  -1.5000E+01   1.5000E+01
18   bimfx_____0029   0.0000E+00   3.5000E-01  -1.9000E+01   1.9000E+01
19   bimfy_____0029   0.0000E+00   1.4000E+00  -1.9000E+01   1.9000E+01
20   bimfz_____0029   0.0000E+00   1.4000E+00  -1.9000E+01   2.9000E+01
```

If `NSET` is larger than real number of sets in the exp. points array, the unused parameters will be fixed, if to use option "f".

7 Summary list of programs

```
HSQ=FUMILIM(A,NAM,NP,EXDA,LEXD,NEXD,US,NIT,OUT,OP,NDIM),
HSQ=FUMIFILES(NAM,NP,HEXD,LEXD,NEXD,US,NIT,OUT,OP,NDIM),
HSQ=FUMHIST(NAM,NP,HEXD,NEXD,ERR,SUB,NIT,OUT,OP),
HSQ=POLIMULT(NPOL,NSET,HEXD,NEXD,FUN,NIT,OUT,OP),
HSQ=FUMREGRES(NAM,NP,US,FILNAM,NEXD,XMIN,XMAX,YMIN,YMAX,FAC),
HSQ=FUMDRIFT(OUTP,EXD,ERR,NP,NDRIFT,NSINGLE,AMIN,AMAX),
CALL MAKEANPW(IAR1,IAR2,CMI,CMA,NC,TMI,TMA,NT,P,C21,IAN),
CALL FUMINI('FILENAME',HNAMES,A,NP),
CALL FUMMULT(HI,HF,NGLO,NLOC,NSET,IST)
```

8 Summary of New features

1. Added intrinsic user subroutines `FUMP0, FUMEXP`, and add-on programs `FUMHIST, POLIMULT, MAKEANPW`.

2. Modernized intrinsic subroutines `FUMGAU, FUMRAY` and add-on program `FUMREG`, including killing of bugs.

3. Modernized approach in subdivision of experimental data into sets. Due to new approach option "g","G" became obsolete and were abolished.

4. Option "f" is abolished. Now the warning follows in case of the presence of unused parameters.

5. Option "w" is modernized. Due to calculation $\langle\chi^2\rangle$ for each set separately, more clear separation of wrong points is achieved for multi-set tasks.

6. The user interface is revised, which is coordinated with a number of users.

7. Due to scrupulous applying of allocate-operators, a lot of memory is saved for multi-set tasks.

8. Subsidiary programs `FUMINI,IFUMMULT` are created, which help to create the initial data file.

9. Simplified the sense of the second parameter(`NAM`) in call to `FUMILIM`. Now, reference to array of names is not provided. Instead, one can use the program `FUMINI`.

9 Testing Programs

The four testing programs are included into the package.

9.1 Program "Thermocouple"

This is a polynomial fit of the real data on calibration of a thermocouple. The program demonstrates efficiency of the Ignoring Wrong Points (IWP) tool. The experimental data are subdivided in sets. Bifurcation is based on the argument value. In

```
hsq=fumifiles('par18.inp',np,'exp.p',3,nex,subarg,nit,'res','W',1)
```

program parameters:
'par18.inp' – reference to the file with initial data;

np – number of parameters in the user's subroutine;
'exp.p' – reference to the file with exp. points;
3 – number of units belonging to one experimental point;
nex – overestimated number of experimental points;
subarg – reference to user's function;
nit – iteration limit;
'res' – name of output file name (different for the standard and parallel fits);
'W' – cut experimental points with huge χ^2;
1 – dimension of the experimental point.

The second call is the same task, but `US` is made heavier to provide consuming time of 10-20 sec., which is comfortable for comparison of the standard and parallel fits. Up to 6 parallel processes are efficient for this task.

9.2 Program "Test3dim"

This program is the simplest polynomial fit of 3-dimensional experimental points. The first call is predestinated for the default structure of experimental points with 9 elements in each point.

```
h=fumifiles('Ai.inp',np,'ex9.dat',9,500,standard,nit,'*','-',3)
h=fumifiles('Ai2.inp',np,'ex6.dat',6,500,user,nit,'*','-u',3)
```

The second call is oriented on the user defined structure of experimental points with 6 elements in each point. As it is seen from value of penultimate parameter, no recording of any files is envisaged. Input data for this program are '*.inp' and '*.dat' files. Output information goes to the screen.

9.3 Program "Rayleigh"

This program demonstrates work with intrinsic subroutines.

The first call is describing momentum distributions of t in $4He$[1]. This spectrum is typical, which can be described within the approach, fixed by Eq.4. The polynomial was chosen as argument of exponent.

The next two calls is describing of data obtained in[2]. The data are described by the sum of Rayleigh and Gauss functions.

The fourth call is connected to data obtained in[3].

One can pay attention that parameter initial values are not defined in these calls. The options, providing tables for theoretical functions is ON in these calls. The results are shown in figs. below.

5-th call fits the regression for $5 \cdot 10^5$ xy-pairs.

Input data for these calls are all *.dat files. Output information is in the 'res' file.

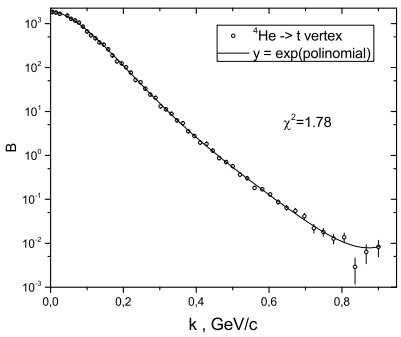

Figure 5: exp(polinomial).

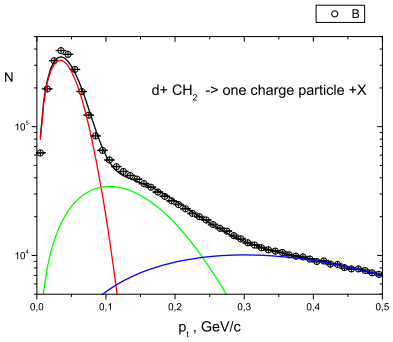

Figure 6: Proton scattering spectrum. Black line – sum of 3 Rayleigh. Color lines - components.

9.4 Program "Evrecons"

This program demonstrates speed option, which can be used when the package is involved in such tasks as event reconstruction. For the linear fit 2 `US` are used:

- no one derivative is defined,
- all derivatives are defined.

The subroutine for the polynomial fit with all defined derivatives is included in the package with name 'fumpol'.

Each call is repeated `ncyc` times (in current version $ncyc = 10^6$). The `NP`, equal to 2 and 3 with 5 exp. points are demonstrated. The first call with reference to definite array of parameter values is always goes by standard

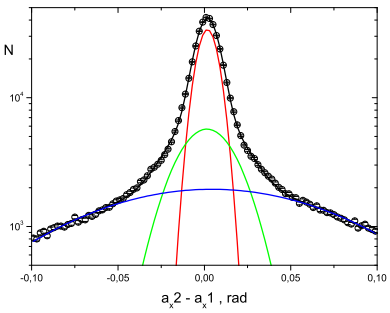

Figure 7: Plane scattering angle distribution. Black line – sum of 3 Gauss. Color lines - components.

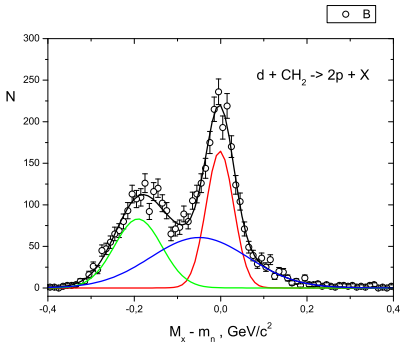

Figure 8: 3 Gauss for two-heads peak. Black line – sum of 3 Gauss. Color lines - components.

branch. The second one depends on conditions, listed in section 5.5. Results of fit (`aini(1:np)`) for first and second call (different branches of the package) of each loop are printed. The parameter initial values (not defined) and experimental points are the same at each call.

For the drift chambers fit with 4 planes the new angle-profile coefficients of the track are generated before each 100 call. Then they are transformed to detector values, then the program reconstructs parameter values. For the first 100 calls comparison of generated and reconstructed values are printed.

For the `NP=2`, the processor 2.8 GHz, Compac 6.6 compiler and 10^6 calls we have:
linear fit without analytical derivatives – 1.45 sec
linear fit with analytical derivatives – 0.92 sec
drift chamber 4 planes – 1.69 sec
drift chamber 4 planes + 1 single detector – 1.86 sec

References

[1] V.G. Ableev et al., Few Body Syst. 8 (1990) 137-144.

[2] S.N. Basilev et al., The XXII Int. Baldin Seminar on High Energy Phys. Problems (2014).

[3] S.N. Basilev et al., J.Phys.Conf.Ser. 678 (2016) no.1, 012040.