ISTITUTO NAZIONALE DI FISICA NUCLEARE

Sezione di Milano

INFN/BE-83/1
1 Marzo 1983

M. Annunziata, E. Binaghi, M. G. Fontaneto, P. Guazzoni, P. Mantica, P. Michelato, A. Pasquinucci, G. Sechi, A. Simonetto and L. Zetta :

SCAMPI - A FORTRAN PROGRAM SIMULATING CHARGE AND MASS PARTICLE IDENTIFICATION

Servizio Documentazione
dei Laboratori Nazionali di Frascati

Istituto Nazionale di Fisica Nucleare
Sezione di Milano

INFN/BE-83/1
1 Marzo 1983

SCAMPI - A FORTRAN PROGRAM SIMULATING CHARGE AND MASS PARTICLE IDENTIFICATION

P. Guazzoni, P. Michelato, G. Sechi[x], L. Zetta
INFN - Sezione di Milano, and Istituto di Fisica dell'Università di Milano

E. Binaghi, M. Annunziata, M. G. Fontaneto, P. Mantica, A. Pasquinucci, A. Simonetto
Istituto di Fisica dell'Università di Milano

ABSTRACT

This paper describes a Fortran program simulating mass and charge ion discrimination, using a SSD telescope, the time of flight technique and different algorithms for MZ^2 identification.

1. - INTRODUCTION

Ion discrimination in mass and charge is a very important problem in light, medium and heavy ion physics. Different solutions have been proposed in literature[1], either hardware or software.

One of the possibilities is to identify the ions in MZ^2, using - for example - a solid state detector telescope and the power-law technique or similar algorithms.

[x] Istituto di Fisica Cosmica e Tecnologie Relative del CNR, Milano.

A simultaneous identification in M, by means of the time of flight technique, allows a full identification of the ion.

In this case the essential part is to choose the best algorithm for the MZ^2 identification and the most suitable ΔE detector thickness, to obtain unambiguous identification in charge and mass.

In order to forecast the final identification, we have written and tested SCAMPI, a Fortran program able to simulate, with different algorithms, the ion discrimination in mass and charge.

2. - GENERAL REMARKS AND DESIGN CONSTRAINTS

In the design of identification systems, the principal difficulty lies in the "a priori" evaluation of the performances of the solution to be chosen. In effect the choice of the algorithm and of the detectors to optimize the identification, may depend on the ions to be detected and on their energies. To minimize these difficulties, we have realized a program with a centralized structure, using a DAISY architecture[(2)] as shown in Sec. 3. This structure, in effect, allows an easy revision of the design avoiding the necessity of rewriting the program completely.

The goal of the designed program is to forecast the goodness of the algorithms and of the experimental solutions adopted, by means of a functional simulation of the whole apparatus.

As a general scheme, to reach a complete simulation of the mass and charge ion discrimination, the program must start from the experimental apparatus data. In effect, once fixed the incident beam and its energy, the target nucleus, the transparent detector thickness, the flight distance, the laboratory angle of ion emission and the parameters of the MZ^2 identification, the program must perform the whole computation for up to ten charges and ten masses for each charge, one hundred reaction products on the whole. In detail, the items to be satisfied for each particle are the following:

a) calculation of the maximum energy allowed by the kinematics and of the minimum energy, corresponding to the cut-off, due to the transparent detector, or to any chosen value;

b) MZ^2 identification with different algorithms ;

c) mass identification;

d) charge identification, combining the results of points b) and c);

e) presentation of the results by means of plots of gaussian distributions.

Moreover, the program must satisfy the constraints of an easy modifiability and an easy full diagnosis possibility.

a) Kinematics

After computing the maximum energy for the ion to be identified, from the usual kinematics formulae, the minimum energy will be computed using the ELOSS program(3), ad hoc modified, to avoid the necessity of rewriting a new energy loss program. Then this energy range will be divided into 39 parts, simulating 39 different energies for the same ion. We have seen(4, 5) that this discretization is enough to forecast correctly the mean behaviour of the on beam experiment. For each energy the program will also calculate the energy lost in the thin detector and its straggling.

b) MZ^2 Identification

The calculation will be performed for all the above energies. As error on the energy lost in the transparent detector, we will use the straggling, propagating it by means of the standard error theory, or we will introduce an "experimental" resolution and perform the calculation with and without this resolution.

To perform the identification we will choose the following algorithms (Particle Identification Function = PIF):

i) the power law(6) : $(E+\Delta E)^x - E^x$;

ii) the multiplier(7) : $(E_o + E + k\Delta E)\,\Delta E$;

iii) the modified power law(8): $\ln(\alpha\Delta E) + (\alpha - 1)\ln(E + c\,\Delta E) - \alpha\ln(300)$.

Following the centralized structure adopted, there is no problem in changing the algorithms adopted without changing the other parts of the program, or in modifying the different sections of the whole program.

c) Mass Identification

In order to perform this calculation, we need to know the time of flight of the product to be identified. Starting from the input data, we must know, for the kinematics computations, the mass of the product. We will use this mass to calculate the 39 times of flight, corresponding to the different particle energies. This is not a vicious circle, because we compute the error on the time of flight, by propagating the energy error, or also in this case we may introduce an "experimental" time spread.

The points a), b), c) will be iterated for all the chosen ions.

Assuming the mean values of MZ^2 and M functions and the errors calculated by means of the propagation formulae, the program will build a gaussian distribution for each case.

d) Charge Identification

This is the final goal of the whole program. During an experiment, the MZ^2 and M identification are coincident, and there is no problem in combining the two previous identifications to obtain the one concerning the charge.

But in a program simulation a lot of memory would be required to ensure the coincidence of different reaction products, as it is necessary to label and remember different energies and time values (in our case a maximum of 11700 values).

To overcome such difficulties, we have chosen a more suitable solution. We can see that the PIF values computed by means of the algorithm i) are proportional to $M^{x-1} Z^{2(9)}$, while the ones computed by means of the algorithm ii) are proportional to $MZ^{2(7)}$.

Using the linear regression method and starting from the calculated PIF and M values, we may fit the best Z value for each ion.

e) Plots of the Results

Two possibilities are required to present the outputs. The first consists in the usual writing of the results, with particular regard to the overlap of the calculated gaussians. The second is the plot of the gaussians simulating the experimental distribution for the peaks corresponding to different ions. Following the known convention, the width of the gaussian distribution is assumed equal to 3σ.

At this point it is possible to insert a spreading factor value to make the peak resolution worse, to obtain a better fit of the experimental results. This spreading factor value is used as a multiplier of the standard 3σ width.

3. - DAISY ARCHITECTURE

In order to cope with the problem of building the "program system" simulating the mass and charge ion identification, the method for designing complex systems, described in ref. (2), has been adopted.

This method consists in organizing the system to be designed with a central structure, called DAISY architecture. Both from the structural and functional point of view, a system structured with a Daisy architecture is composed of two subsystems :

a) A control subsystem, which performs the whole decisional activity of the system and, on the basis of an external input, provides for the activation of the second subsystem ;

b) A controlled subsystem, consisting of several independent modules, which are allowed to communicate only with the control subsystem. Each of these modules accomplishes only one particular elementary action among those which the system has to carry out.

The activity of the control subsystem can be described in terms of a function whose arguments are the external input at time (T) and the state of the modules activated at time (T-1).

This activity consists of:

- reading the input information from the environment;
- interpreting such information;
- selecting and activating one of the controlled subsystems;
- reading new information from the environment.

Therefore, the working scheme of the whole system consists of a cyclic recognize-act paradigm; in fact, a new cycle starts only when a new command is provided from the environment. The whole process ends when the proper end-of-execution command is supplied.

This kind of architecture - i. e. Daisy - is characterized by the following important properties:

- the controlled subsystems, being independent of the previous state, allow an easy full diagnosis;
- since the controlled modules can communicate only with the control system, one is given the opportunity to test each component separately, so increasing the chance of locating and detecting any error in the system;
- the complexity of the system is reduced, because it is shared between the controlled modules (which can be activated at the same time) and the activation sequence produced by the control system.

The fact that the complexity of the system is so rigorously spread over a horizontal structure (the set of parallel modules) and a vertical one (the activation sequence), makes the whole system flexible and easy to diagnose.

This pattern provides, therefore, the best instruments to face the problem of correctness in its most critical point, i. e. in verifying the correct way of real system functioning, since it is impossible to forecast it during the design.

This methodology has been applied in order to create SCAMPI as a "program system". In this case it must be composed of:

- a central controller with functions of getting the external commands, checking their correctness and the availability of the required actions, selecting the corresponding action among the N available;

- several modules designed to accomplish the elementary actions, which can be selected and activated by the central controller.

The set of instructions to check the correctness, to select among the N possibilities and to activate the action, is inserted in a loop, which ends only when the user provides the correct commands.

In this way, the execution sequence of the program is not "a priori" fixed, but evolves dinamically, depending on the user's commands. The subroutines performing the elementary actions of the system can be user defined or system modules. The first ones depend on the particular problem to be solved by the Daisy program. Conversely, the second ones have standard structure. Usually they are independent of the problem and consist of error processing, input and output modules.

4. - PROGRAM DESCRIPTION

4. 1. - Program Structure

In order to provide the program with facilities for an easy modifiability and diagnosis, the DAISY architecture was adopted for it (as shown in the previous section).

SCAMPI is therefore structured in this way (Fig. 1a):

1) The main program, which calls a "central controller" as a subroutine. The central controller is called in a loop, which ends when the main program is given back an "end of execution" code.
2) The central controller, which performs the initialization as first action. Therefore, the diagnostic messages and the symbolic commands are read by the proper modules from an external file. Afterwards, the central controller proceeds to the activation of the endmodules, repeating the following steps :
   a) the input-output module SPV1 is activated, to read the user's symbolic command;
   b) the "translator" module FIND is then activated, to translate the user's command into numerical form ;
   c) then the proper module is activated, to perform the action which was requested by the user at a). The execution of the user's command may sometimes require the sequential activation of several modules ;
   d) in case of errors, the GER module, which deals with error processing, is put in action.

   Execution ends (that is, the main program is given back an end-of-work code) when the user provides the proper "end-of-work" command.

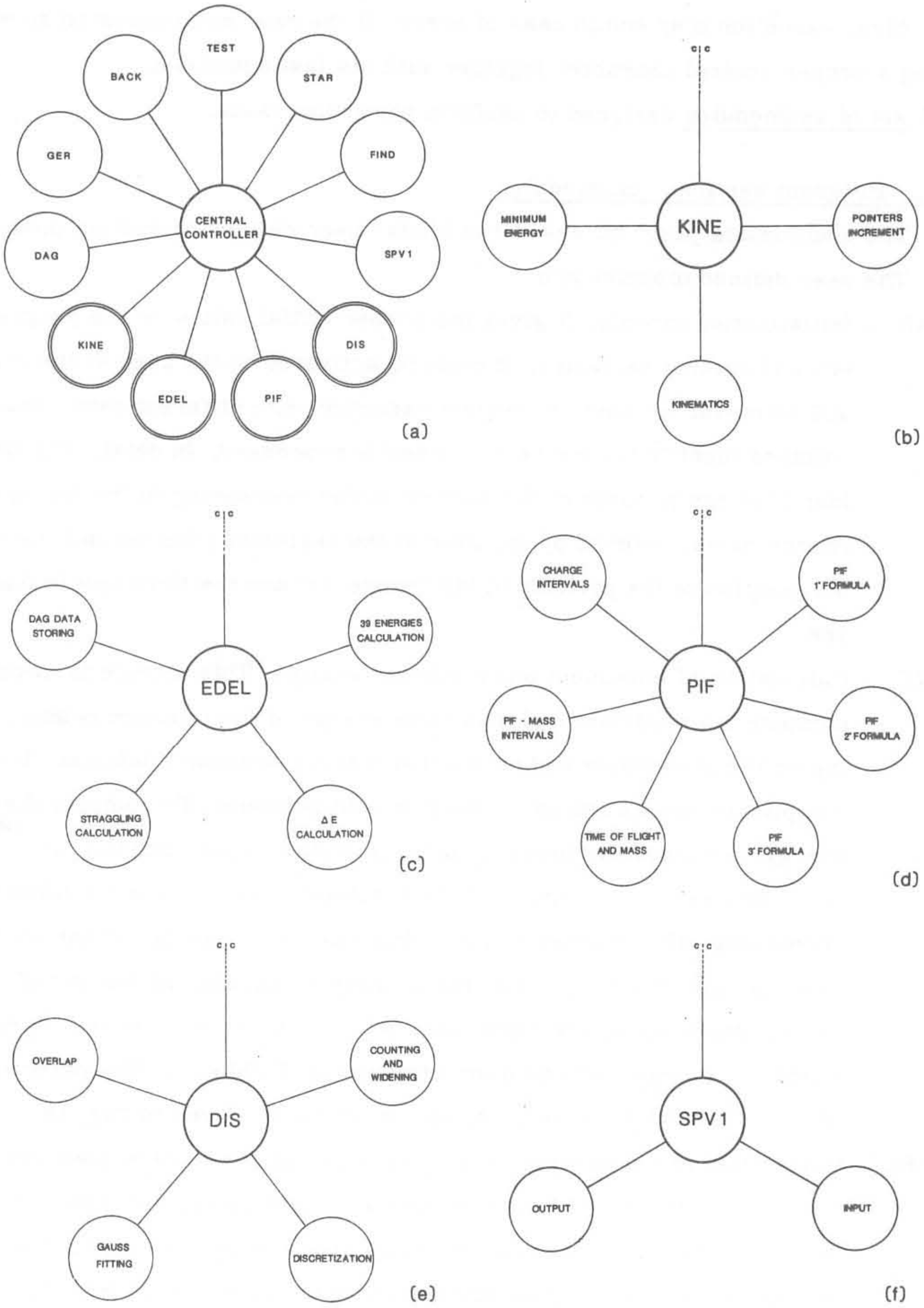

FIG. 1 - Generalized block diagrams of the main sections of SCAMPI: a) central controller with endmodules (the double circled are computing endmodules); b) endmodules KINE with its controlled modules; c) endmodule EDEL with its controlled modules; d) endmodule PIF with its controlled modules; e) endmodule DIS with its controlled modules; f) endmodule SPV1 with its controlled modules.

Also, execution may end in case of error, if the user has requested so by entering a proper control character together with his last command.

3) A set of endmodules designed to perform particular tasks.

4. 2. - Program sections : endmodules

The endmodules provided are of two kinds : user defined or system modules.

The user defined modules are :

STAR - Initialization module. It gives the proper initial values to the program pointers and control variables. It must be activated by the user at the beginning, and whenever he wants to restart execution with different data. The program pointers identify the ion being currently processed. In detail : the first one identifies the position of the particle under processing in the list of mass and charge pairs, entered by the user at the beginning; the second one identifies the position of the particle in the charge list and the third one in that of masses.

KINE - Calculation of maximum and minimum energy. This module is designed to compute the minimum and maximum energy of the reaction product, depending on the kinematics and on the transparent detector thickness. It also accomplishes the increment of the program pointers. To compute the minimum energy, a successive approximation method is used, starting with the mean value between zero energy and the maximum energy, and calculating the corresponding ΔE, in order to see if this energy is enough for the particle to pass the thin detector. If so, the process is iterated on the left of the initial value, otherwise on the right, until a value is reached, which represents the minimum energy, with an approximation of ± 100 keV. The partition of the present endmodule in its controlled modules is shown in Fig. 1b.

EDEL - Calculation of the energies lost in the thin detector and of their errors. As seen above, the 39 particle energies are sufficient to simulate on beam experiment. For each of these 39 values, the energy loss in the thin detector and the per cent straggling are computed by means of the modified ELOSS program. The maximum of the so obtained straggling values is taken as the real one. The partition of the present endmodule in its controlled modules is shown in Fig. 1c.

PIF - Calculation of the PIF, mass and charge functions with relative errors. The PIF and its standard deviation are computed for each of the 39 energies, using one of the three algorithms shown in Sec. 2b), according to the user's choice. A weighted average among the 39 PIF values and its standard deviation are then calculated. The time of flight and its per cent error are also computed. The above values are used to obtain the measured masses and their errors, by means of the following formulae:

$$\overline{M} = \frac{\bar{\tau}^2 E}{A^2 s^2} + \frac{M}{4E^2}\,\sigma_E^2 ,$$

$$\sigma_M^2 = \frac{\bar{\tau}^4}{A^4 s^4}\,\sigma_E^2 + \frac{4\bar{\tau}^2 E^2}{A^4 s^4}\,\sigma_\tau^2 ,$$

where:

A = 0.6946
M is measured in amu
E is measured in MeV
τ is measured in ns
s is measured in cm

and propagating the errors on energy and time.

The slope, intercept and linear regression coefficient of the PIF vs MZ^2 (or $M^{x-1}Z^2$) plots are then computed. These and all the above results are employed to obtain the mean value and the standard deviation of the charge for each particle. The partition of the present endmodule in its controlled modules is shown in Fig. 1d.

DIS - Range discretization and curve fitting. To provide easily interpretable forecasts it is necessary that the final output is a plot of the significant results vs channel number (up to 4096). Gaussian curves with 3σ spread or more, depending on the spreading factor value, are then built on the intervals resulting from the above processing, to simulate the peaks of the on beam spectra. The user can choose to perform the above actions either on the intervals concerning PIFs, masses and charges of all the processed ions, or on those concerning the identification functions relative to the isotopes currently processed. In the latter case the overlap percentage is also calculated. The partition of the present endmodule in its controlled modules is shown in Fig. 1e.

DAG - This module allows the printing and the punching out on paper tape of the 39 pairs of E, ΔE and of the 39 time of flight values for each ion, to prepare the input data for the DAG pulser(4) ad hoc modified to allow up to four coincident output pulser.

TEST - Data testing module. This module performs, on input data, all the tests required to ensure their consistence with the program specifications.

BACK - Reset of program pointers. This module resets the program pointers one step back, in order to allow the reprocessing of the last ion.

The system modules are:

GER - Error processing module. It prints a diagnostic message for every error occurring anywhere in the program. During initialization phase, it loads the diagnostic messages from an external file and, since then, it can print out the complete list of messages, whenever the user requests it.

FIND - Translation module. This module performs the translation of the user's symbolic commands into numerical form. This action is required because the symbolic commands are defined by the user and are therefore non-standard. The translation is accomplished on a merely positional basis.

SPV1 - Input-output module. It runs two other modules: the input one and the output one. It can display all the input data entered by the user at the beginning and any other significant computed quantity. It also attends to the printing of plots. This module allows the user to replace the computed values of minimum energy and straggling, depending on the experimental constraints. The partition of the present endmodule in its controlled modules is shown in Fig. 1f.

4. 3. - Technical data

SCAMPI is written in Fortran IV language. It consists of 2605 statements. In addition to the main program there are 92 subroutines.

The program can run either in batch mode or interactively. SCAMPI usually runs on the Siemens 7865 computer with IBM VM operative system of the S. I. A. M. center of the C. N. R. in Milano. The memory required during running is 231 kB.

The test case shown in the next section (forty two ions) has employed 69 minutes of virtual C. P. U. time.

The bottle-neck of the program consists in the calculation of the 39 energies lost by each ion in the transparent detector. The mean virtual C. P. U. time for computing each energy loss set of 39 values, and relative minimum energy is about 1 minute.

For sake of simplicity the plots are made with the line printer, to combine them with the numerical data. It is trivial the use of a plotter.

The data to be entered at the beginning are:

- title;
- flight distance;
- energy of the incident beam;
- mass of incident ions and of target nucleus;
- thickness of thin detector;

- mass and charge of each reaction product to be processed;
- Q-values of the reactions to be considered;
- adjustable parameters for PIF computing.

The data which can be displayed on user request at execution time are:

- symbolic commands list;
- diagnostic messages list;
- input data;
- current values of program pointers;
- minimum energy;
- per cent straggling;
- times of flight and their errors;
- PIF and mass relative to the ion under processing;
- PIFs, masses and charges relative to all processed particles;
- overlap percentage among the PIF, mass and charge intervals for the element under processing;
- parameters of linear regression used to calculate charges;
- plots and numerical values of the results.

Moreover, the following actions can be carried out, if required:

- replacement of the minimum energy;
- replacement of the per cent straggling;
- insertion of the thin detector energy resolution;
- insertion of the time resolution for the time of flight;
- correction of the input data.

The list of the available symbolic commands is the following:

INPU - To supply the input data, which are given by the user after suitable request messages;

SEIN - To display the input data in order to check their correctness;

CORR - To allow the user correction of possible mistakes made during the input phase;

STAR - To initialize the pointers and control parameters. It must be provided once at the beginning of the execution, or whenever one wants to restart the whole computation;

TEST - To test the input data to verify that they are within the correct range;

RISO - To allow the user to choose between the use of the error theory and the use of an experimental resolution for the calculation of MZ^2 and mass functions;

KINE - To calculate the maximum and minimum energy of each ion and to accomplish the increment of program pointers;

EDEL - To calculate the 39 values of the energy lost in the thin detector;

EMIN - To display the calculated values of the maximum and minimum energy;

STRG - To display the straggling value;

EINP - To replace the calculated minimum energy with an input value;

STIN - To replace the calculated straggling with an input value;

PIF1
PIF2
PIF3 - To calculate PIFs, times of flight, masses and charges with their errors, using for the PIF calculation one of the three selected formulae;

TIME - To display the 39 energy losses and the 39 times of flight with their percent errors;

DAG* - To punch out the E, ΔE and τ values for the DAG pulser;

TPIF - To display the PIF ranges concerning all the processed particles;

TMAS - The same as above, for the masses;

TOTZ - To display the charge ranges and the slope, intercept and linear regression coefficient of the PIF vs MZ^2 (or $M^{x-1}Z^2$) line;

INTP - To display the PIF and mass ranges concerning the ion under processing;

RANG - To provide the number of channels on which discretization will be made;

DIS1 - To accomplish discretization of mass ranges at fixed ion. To build gaussian curves over them and to calculate the overlap percentage;

DIS2 - The same as above, for the PIF;

DIS3 - To accomplish discretization of charge ranges of all the ions and build the gaussians;

DIS4 - The same as above, for the PIFs;

DIS5 - The same as above, for the masses;

SUPR - To display the calculated overlap percentage between the gaussian curves;

GRAP - To display the graph of the gaussian curves, built as shown before;

BACK - To reset the pointers values one particle back, in order to repeat the calculation for that particle;

WHAT - To give information about the elaboration state, by showing the current values of the program pointers;

DUMP - To provide a dump of the input data and of the main calculated values;

SEFR - To list all the error messages by displaying the content of the two diagnostic tables;

HELP - To display the list of symbolic commands, with the explication of their functions.

END* - To terminate the execution.

5. - PROGRAM RESULTS

To test SCAMPI, we have simulated many reactions, especially for the MZ^2 identification, for which we have experimental results(5).

In this paper we present only one test case, up to the charge identification. It is an analysis of the reaction induced by a 261 MeV ^{20}Ne beam on ^{232}Th(8), at $\theta_{lab} =$ $= 25^o$ laboratory angle. The emitted ions, considered in this test are:

Li	: 6, 7, 8, 9
Be	: 7, 9, 10
B	: 10, 11, 12, 13
C	: 11, 12, 13, 14, 15, 16
N	: 12, 13, 14, 15, 16, 17
O	: 15, 16, 17, 18, 19, 20
F	: 17, 18, 19, 20, 21
Ne	: 20, 21, 22, 23
Na	: 21, 22, 23, 24

Their total number is 42: 9 charges and up to 6 masses for each charge. The PIF algorithm used is the power-law formula.

The transparent detector thickness is assumed to be 100 μm, with proper straggling as resolution and the flight distance 10 cm. The exponent for the power-law formula is assumed $x = 1.6$. The method used is error propagation.

In Figs. 2, 3 and 4 the final plots of PIF, M and Z distribution are shown. As it can be seen, the results allow an easy forecast of the total discrimination in mass and charge and also confirm the goodness of the $M^{x-1}Z^2$ fitting for the power-law identification, as referred in ref. (9). In fact the linear regression results for the above fitting are:

- linear regression coefficient : $r = 0.99972$
- slope : $a = 5.0847$
- intercept : $b = 35.7134$.

To conclude we think that SCAMPI is a good instrument during the design of identification systems or the preparation of experiments.

We express our acknowledgments to Dr. s P. Boccaccio and G. Viesti, for having made available the Eloss program.

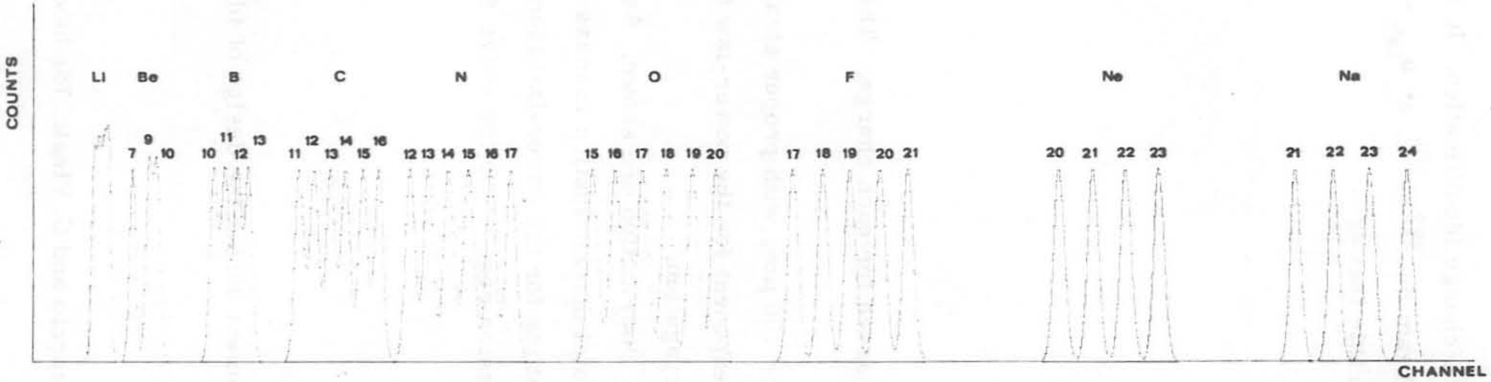

FIG. 2 - Plot of the PIF distribution for all the ions referred in the text (spreading factor value = 8; power-law exponent x = 1.6).

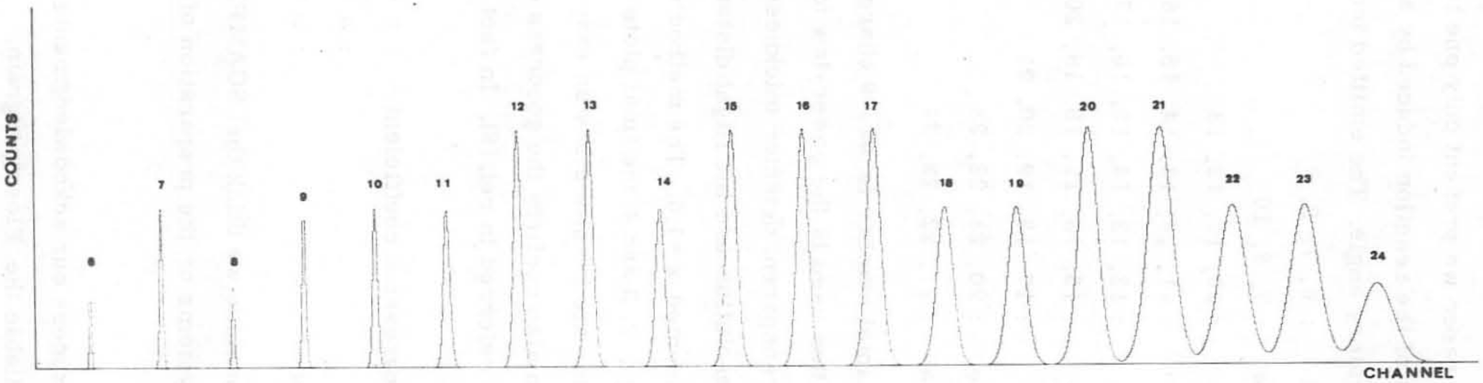

FIG. 3 - Plot of the M distribution for all the ions referred in the text (spreading factor value = 4).

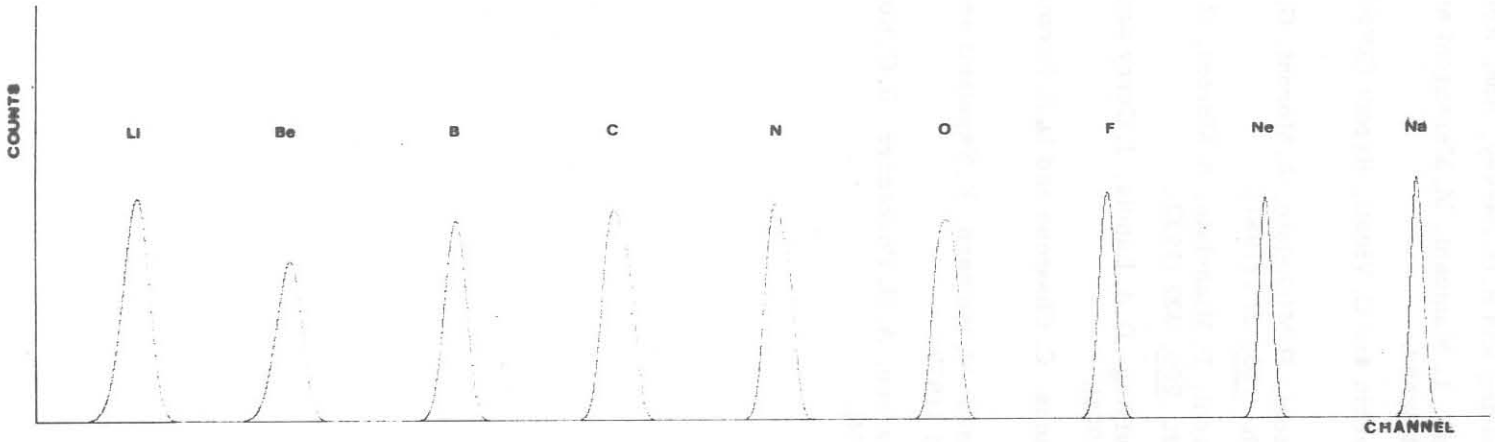

FIG. 4 - Plot of the deduced Z distribution, using the linear fitting PIF vs. $M^{x-1}Z^2$. The obtained coefficients are shown in the text (spreading factor value = 4).

REFERENCES

(1) - F. S. Goulding and B. G. Harvey, Ann. Rev. Nucl. Sci. 25, 167 (1975).

(2) - E. Binaghi, L. Mantoani, M. Marangoni and G. Sechi, private communication and to be published.

(3) - P. Boccaccio and G. Viesti, Report INFN/BE-80/1 (1980), and private communication.

(4) - P. Guazzoni, P. Michelato, A. Moroni, G. F. Taiocchi and L. Zetta, Nuclear Instr. and Meth. 185, 219 (1981).

(5) - P. Guazzoni, P. Michelato, A. Moroni, G. F. Taiocchi and L. Zetta, Nuclear Instr. and Meth. 200, 323 (1982).

(6) - F. S. Goulding, D. A. Landis, J. Cerny and R. E. Pehl, Nuclear Instr. and Meth. 31, 1 (1964).

(7) - M. W. Sachs, C. Chasman and D. A. Bromley, Nuclear Instr. and Meth. 41, 213 (1966).

(8) - T. Shimoda, M. Ishihara, K. Nagatami and T. Nomura, Nuclear Instr. and Meth. 165, 261 (1979).

(9) - J. D. Bowman, A. M. Poskanzer, R. G. Korteling and G. W. Butler, Phys. Rev. C9, 836 (1974).