The Fitter Handbook

Alexei Soloviev[1], Alexei Stadnik, Akhmed Islamov and Alexander Kuklin
Version: 2.1.0 (2007/11/22)

Fitter is a C++ program aimed to fit a chosen theoretical multi-parameter function through a set of data points. The method of fitting is chi-square minimization. Moreover, the robust fitting method can be applied in Fitter. Fitter was disigned to be used for a small-angle neutron scattering data analysis. Respective theoretical models are implemented in it. Some commonly used models (Gaussian and polynomials) are also implemented for wider applicability.

Fitter web site: `http://www.jinr.ru/programs/jinrlib/fitter/indexe.html`

1 Introduction

Fitter, the package for fitting a chosen theoretical multi-parameter function through a set of data points, has been successfully used in small-angle neutron scattering data analysis since 2003 [1]. The several examples of researches using Fitter are given below. This handbook gives a long write-up description of the new version of Fitter.

The small-angle neutron scattering (SANS) method has long history. So, the earliest references to SANS can be traced back to 1948 by D.J.Huges et al. [2]. Starting with classical work of A.Guinier, who suggested the so called "Guinier approximation" to powerfull package of D.Svergun team [3, 4, 5, 6, 7, 8, 9], who developed and brogth SANS method up to present moment. At the same time, in spite of the wide SANS method application and already more than 50-years history of its development, the problem of interpreting the data obtained on spectrometers, still remain open. It is related, first of all, to the ill-posedness of the solved problem and, on the other hand, to specifics of existing SANS spectrometers. Practically each spectrometer is provided by not only software on initial processing experimental data, but also by a complex of programs for fitting the curves measured in coordinates scattering intensity/q, where $q = (4\pi/\lambda)\sin(\theta/2)$.

A modernization of the time-of-flight SANS YuMO spectrometer at the high flux pulse IBR-2 reactor (JINR, Dubna) [2] was launched in 2000. One of the goals of the improvement was to increase the dynamic Q-range of the instrument. It was realized by using a two-detector system with central holes at the YuMO instrument. For this purpose "old" and "new" detectors were used. The main view of the setup of the modernized YuMO instrument as well as an example of a SANS experiment are presented in [10]. The use of the two detectors YuMO set up for the users have clearly demonstrated its high efficiency [11, 12, 13, 14, 15, 16, 17, 18, 19, 20]. That are only several examples. In 2006 will start at the YuMO spectrometer position sensitive detector [21].

The new Fitter version is better suit requirements of new detector setup. The main changes are:

[1]E-mail: solovjev@cv.jinr.ru

- fitting in a given range;
- improving the robust approach;
- choosing the MINUIT strategy;
- exponential function in STD models;
- new visualization and GUI;

This paper is a long write-up description of the current version of program for fitting a chosen theoretical multi-parameter function through a set of data points. Examples of application to experiments on the small-angle spectrometer YuMO are considered.

2 Long Write-Up

The Fitter is designed to be used for SANS data processing first of all. Thus, SANS theoretical models are implemented in it. Moreover, some standard mathematical models are added for wider applicability. Besides the implemented theoretical models, Fitter has a minimization module. It provides a safe call of MINUIT [25] procedures in the current version. The important feature of Fitter's design is its expandability: both new models and new minimizing algorithms can easily be added to the existing ones.

2.1 Theoretical Models

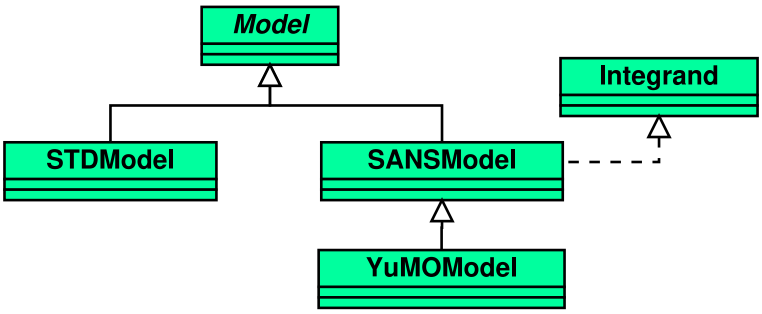

Fig. 1: Fitter's model module class hierarchy

Fitter's model module is designed as follows (fig.1). Abstract base class `Model` provides a common interface, used by a minimization module. Thus, any theoretical model class inherits from it. Model classes currently implemented in the Fitter are: `STDModel`, `SANSModel`, `YuMOModel`. Each concrete class provides several theoretical functions. All of them are described below.

2.1.1 Standard Models

Standard mathematical models implemented in the Fitter are:

- **Gaussian**

$$G(x) = A\exp\left\{-\frac{1}{2}\left(\frac{x-x_0}{\sigma}\right)^2\right\} + B$$

- **Exponential**

$$E(x) = A\exp\{mx+b\} + B$$

- **Polynomials** up to the eighth order

$$P_n(x) = \sum_{k=0}^{n} p_k x^k$$

2.1.2 SANS (Small-Angle Neutron Scattering) Models

Determination of invariants for small-angle scattering curves allows one to analyze the structure of a particle under study. Upon the first step of this analysis the particle form is approximated by simple geometrical bodies - ellipsoids, cylinders, prisms.

Thus, SANS models implemented in the Fitter are:

- **Ball** of the radius R

$$\begin{aligned} I(Q) &= A\Phi^2(QR) + B \\ \Phi(t) &= 3\frac{\sin t - t\cos t}{t^3} \end{aligned}$$

- **Ellipsoid of revolution** with half-axises a and av

$$\begin{aligned} I(Q) &= A\int_0^1 \Phi^2\left[Qa\sqrt{1+x^2(v^2-1)}\right]dx + B \\ \Phi(t) &= 3\frac{\sin t - t\cos t}{t^3} \end{aligned}$$

- **Cylinder** of the radius R and the length H

$$\begin{aligned} I(Q) &= A\int_0^1 \Lambda_1^2\left(QR\sqrt{1-x^2}\right)S^2(QHx/2)dx + B \\ \Lambda_1(t) &= 2J_1(t)/t \\ S(t) &= \sin t/t \end{aligned}$$

- **Elliptical cylinder** with half-axises a, av and the length H

$$
\begin{aligned}
I(Q) &= A\int_0^1 \Psi_{EC}(Q, a\sqrt{1-x^2})S^2(QHx/2)dx + B \\
\Psi_{EC}(Q,a) &= \frac{1}{\pi}\int_0^{\pi} \Lambda_1^2\left(Qa\sqrt{\frac{1+v^2}{2}+\frac{1-v^2}{2}\cos y}\right)dy \\
\Lambda_1(t) &= 2J_1(t)/t \\
S(t) &= \sin t/t
\end{aligned}
$$

- **Parallelepiped** with the ribs a, b, c

$$
\begin{aligned}
I(Q) &= A\int_0^1 \Psi_p\left(Q, b\sqrt{1-x^2}, a\right)S^2(Qbcx/2)dx + B \\
\Psi_p(Q,b,a) &= \frac{2}{\pi}\int_0^{\pi/2} S^2(Qa\sin y/2)S^2(Qb\cos y/2)dy \\
S(t) &= \sin t/t
\end{aligned}
$$

- **Spherical shell** of the outer radius R_1 and the inner radius R_2

$$
\begin{aligned}
I(Q) &= A\left[\Phi(QR_1) - \left(\frac{R_2}{R_1}\right)^3\Phi(QR_2)\right]^2 + B \\
\Phi(t) &= 3\frac{\sin t - t\cos t}{t^3}
\end{aligned}
$$

More detailed information about SANS models is available in [22].

2.1.3 SANS Models with YuMO spectrometer resolution

These models are implemented to fit data measured on the YuMO spectrometer [23] operated on the 4-th channel of the fast pulsed reactor IBR-2 [24]. YuMO models are the same as SANS, but they take into account the spectrometer resolution.

- **SANS Model with resolution**

$$
\begin{aligned}
I(Q) &= \frac{1}{\sqrt{2\pi}\sigma}\int_{Q-3\sigma}^{Q+3\sigma} I_{SANS}(q)\exp\left\{-\frac{1}{2}\left(\frac{q-Q}{\sigma}\right)^2\right\}dq \\
\sigma &= \sqrt{2\Delta Q}
\end{aligned}
$$

2.2 Minimization

To find theoretical model parameters, one should minimize a functional, which is a measure of deviation between a theoretical curve and experimental data. In a common case of a least-squares fit, the functional under minimization is a chi-square:

$$\chi^2 = \frac{1}{N - N_{parms}} \sum_{i=1}^{N} \left(\frac{f(x_i) - y_i}{\Delta y_i} \right)^2$$

2.2.1 Minuit

We are using ROOT::TMinuit class to perform a minimization. This package was originally written in Fortran by Fred James and part of PACKLIB (patch D506) and has been converted to a C++ class by R.Brun. The current implementation in C++ is a straightforward conversion of the original Fortran version. The main changes are:

- The variables in the various Minuit labeled common blocks have been changed to the TMinuit class data members.
- The internal arrays with a maximum dimension depending on the maximum number of parameters are now data members arrays with a dynamic dimension such that one can fit very large problems by simply initializing the TMinuit constructor with the maximum number of parameters.
- The include file Minuit.h has been commented as much as possible using existing comments in the code or the printed documentation
- The original Minuit subroutines are now member functions.
- Constructors and destructor have been added.
- Instead of passing the FCN function in the argument list, the addresses of this function is stored as pointer in the data members of the class. This is by far more elegant and flexible in an interactive environment. The member function SetFCN can be used to define this pointer.
- The derived class TMinuitOld contains obsolete routines from the Fortran based version.

Additional modifications were made to separate TMinuit class from the ROOT package (Fitter is ROOT independent indeed).

MINUIT offers the user a choice of several minimization algorithms. The MIGRAD algorithm is, in general, the best minimizer for nearly all functions. It is a variable-metric method with inexact line search, a stable metric updating scheme, and checks for positive-definiteness. Its main weakness is that it depends heavily on knowledge of the first derivatives, and fails miserably if they are very inaccurate.

For further details see MINUIT documentation [25].

2.2.2 Robust Fitting

The least-squares fitting involves the minimization of the sum of the squared residuals. There are two instances where this minimization produces less than satisfactory fit:

1. The significant outliers are present in data. In this case, the square of the residuals of these out-lier points may, within a given region, significantly shift the fitted curve away from the bulk of the data.
2. The the Y-data span more than several orders of magnitude. The squared residuals of the largest valued Y-points can overwhelm the influence of the squared residuals of the smallest Y-valued points, causing the smallest Y-value points to either be poorly fitted or not fitted at all. Data that require a logarithmic Y-scale to see all of the points may be a good candidate for robust fitting, especially if four or more major log divisions take place.

Robust estimates designed to be successful in such cases. The essence of robust fitting is to use a minimization that is less influenced by outliers and the dynamic range of the Y-variable. Each data measured point took into account with it own weight, which indicate influence of given point.

It is based on so-called M-estimates which follow from maximum like-hood approach, M-estimates are usually the most relevant for model fitting. Robust approach uses gross-error model (Hubert). Probability distribution function of measurement errors d_i suggested as a superposition of two distributions: basic $g(d_i)$ and distribution of big errors $h(d_i)$

$$
\begin{aligned}
D(d_i) &= (1-\epsilon)g(d_i) + \epsilon h(d_i), \quad \epsilon \in [0,1] \\
g(d_i) &= \frac{1}{\sigma\sqrt{2\pi}} \exp\left\{\frac{-d_i^2}{2\sigma^2}\right\} \\
h(d_i) &= \alpha \exp\{-\beta|d_i|\}
\end{aligned}
$$

In practice, robustness works also in cases of other distributions $h(d_i)$, for example, normal one like $g(d_i)$ is, but with bigger RMS.

Application of maximum like-hood approach and few simplifications and the fact that RMS of the distribution $g(d_i)$ can be approximated by chi-square of data points, give us well known Tukey bi-square weights, which is suitable in most cases.

$$
w_i = \begin{cases} \left(1 - \dfrac{|f(x_i) - y_i|^2}{c_T\chi^2}\right)^2, & |f(x_i) - y_i| \le c_T\chi^2 \\ 0, & |f(x_i) - y_i| > c_T\chi^2 \end{cases} \tag{1}
$$

In chi-square function data values are multiplied by their weights:

$$\chi^2 = \frac{1}{\sum\limits_{i=1}^{N} w_i - N_{parms}} \sum_{i=1}^{N} w_i \left(\frac{f(x_i) - y_i}{\Delta y_i} \right)^2 \tag{2}$$

The initial values of the weights are equal to 1. In the following iterations, weights are recalculated after each procedure of minimization and calculation of the new value of chi-square. Iterations are repeated until convergence (until chi-square value is stabilized within a predefined accuracy).

In general, the robust approach looks like:

- setting $w_i = 1$, calculating χ^2 by (2), setting $c_T = 5$;
- while $c_T > 1$ doing the loop:
  - calculating w_i by (1) using current values of χ and c_T;
  - minimizing χ^2 given by (2);
  - reducing c_T by $c_T / = 2$.

For detailed explanation see [26, 27, 28].

3 Installation

3.1 Installation on UNIX systems

In order to compile and install Fitter on your system, type the following in the base directory of the `Fitter` distribution:

```
% ./configure
% make
% make install
```

To build `Fitter` with GUI you should install FLTK [29] first. Second, you should install `Gluplot` ploting package [30], since it is used for visualization. Finally, enable using FLTK libraries while `Fitter` configuring:

```
% ./configure --enable-fltk
```

Since `Fitter` uses `autoconf` you should have not trouble compiling it. Should you run into problems please report them to the the authors Alexei Soloviev and Alexei Stadnik.

3.2 Installation on Windows

`Fitter` provides an user-friendly setup procedure for Windows users. Just run `setup` and follow the instructions.

4 Usage

4.1 General Usage

Fitter is a C++ program aimed to fit a chosen theoretical multi-parameter function through a set of data points. The method of fitting is chi-square minimization. Moreover, the robust fitting method can be applied in Fitter. Fitter was designed to be used for a small-angle neutron scattering data analysis. Respective theoretical models are implemented in it. Some commonly used models (Gaussian and polynomials) are also implemented for wider applicability.

4.1.1 SYNOPSIS

fitter **–{ std| sans| yumo}** *model* [**-s** *name*] [**-r**] [**-R** *x1:x2*] [**-v**[*file*]] [**-o** *file*] [**-H** *file*] [**–rc-read**] [**–rc-write**] *file*

fitter [**-h**] [**-V**]

4.1.2 INPUT

Input data file (ASCII) for fitter should have at least three columns: the first one would be interpreted by fitter as ”X”-samples, second one - as ”Y”-samples and the third one - as ”Y”-errors. For YuMO models (see below) should be one more column, which is interpreted as ”X”-errors.

4.1.3 OPTIONS

MODEL OPTIONS

–std ***model*** Standard models. One of the following (see '–std help' for list):

- *g* Gaussian
- *e* Exponential
- *p0* Polynomial of the order 0
- *p1* Polynomial of the order 1
- *p2* Polynomial of the order 2
- *p3* Polynomial of the order 3
- *p4* Polynomial of the order 4
- *p5* Polynomial of the order 5
- *p6* Polynomial of the order 6
- *p7* Polynomial of the order 7
- *p8* Polynomial of the order 8

–sans ***model*** SANS models. One of the following (see '–sans help' for list):

- *b* Ball
- *c* Cylinder

c2 Elliptical cylinder
e2 Two axis ellipsoid
p Parallelepiped
ss Spherical shell

–yumo *model* SANS models for YuMO spectrometer. One of the following (see '–yumo help' for list):

b Ball with resolution
c Cylinder with resolution
c2 Elliptical cylinder with resolution
e2 Two axis ellipsoid with resolution
p Parallelepiped with resolution
ss Spherical shell with resolution

PROCESSING OPTIONS

-s, –strategy *name* Change MINUIT strategy. MINIMIZE is a default one.

-r, –robust Robust fitting. Useful for highly noised data.

-R, –range *x1:x2* Fitting in range from *x1* to *x2*. Note that no fit is applied if a range appearers to be empty.

-v, –verbose[*file*] Output MINUIT processing information. No output is produced by default. Repeating this option determines how much output will be produced:

1. minimum MINUIT output
2. normal MINUIT output
3. additional output giving intermediate results
4. maximum output, showing progress of minimizations

OUTPUT OPTIONS

-o, –output *file* Output data file with the resulting theoretical curve. No file is created by default.

-H, –header *file* Output resulting parameters and chi-square value to file (standard output by default). File may be the same as for theoretical curve, parameters and chi-square will be put to its beginning in this case.

–rc-read Read rc-file. If specified, this option cause reading parameters staring values from rc-file. Some times this can highly speed-up minimization. Rc-file `.fitter` should exist in $HOME (in C: on Windows).

–rc-write Write rc-file. If specified, this option cause writing resulting parameters to rc-file. Useful for creating/repairing rc-file. Rc-file `.fitter` is placed in $HOME (in C: on Windows).

HELP OPTIONS

-h, --help Display short help message and exit.

-V, --version Display version information and exit.

4.2 GUI

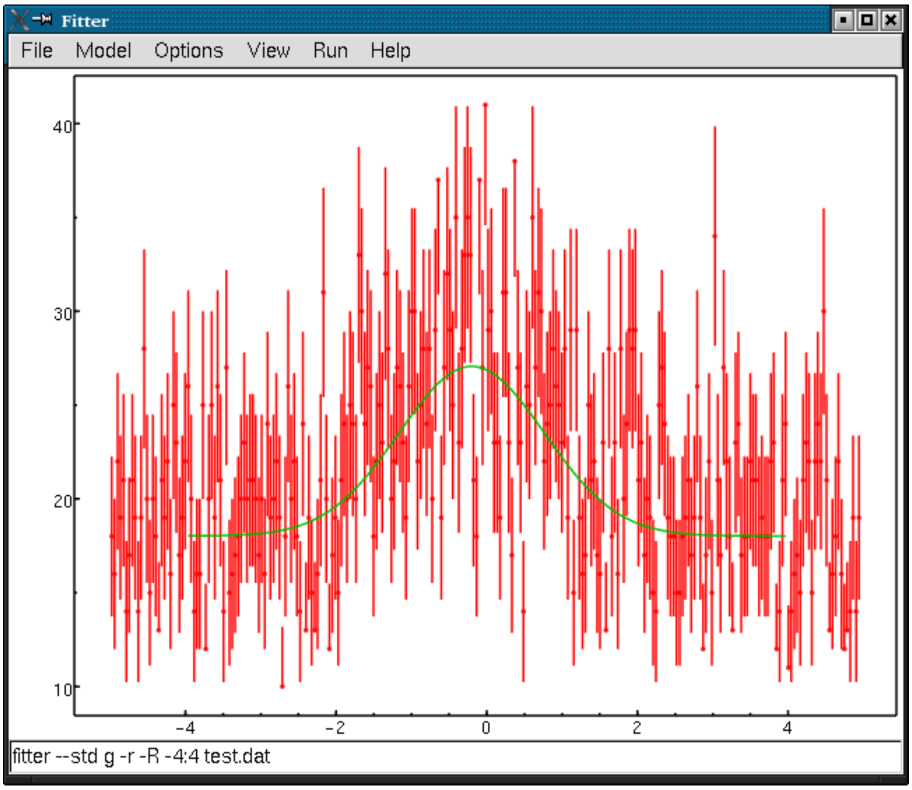

Fig. 2: Fitter's main window

Besides the command-line interface the Fitter offers an user friendly GUI (graphical user interface). Fitter's main window, shown in fig.2, is implemented in the standard manner with menu-bar, tool-bar and status-bar. All menu items exactly correspond to the command-line options described above. The command-line to be executed is just formed by choosing corresponding menu items and is displayed on the status-bar. Some of the most frequently used menu items are duplicated as buttons or combo-boxes on the tool-bar. A window with fitting results arises after fit is complete.

5 Examples

5.1 Robust feature example

Here is a simple example of the Robust feature usage. File `test.dat` is an example of input data for fitter. It is just a generated gaussian distribution, contaminated by uniformly distributed random noise.

Typing in command line

```
% fitter test.dat -o fit1.dat --std g
```

we get the following result:

```
----------------------------------------------------
Model.......................................Gaussian
----------------------------------------------------
Mean............................-0.4576 +- 0.1719
RMS..............................1.7571 +- 0.3680
Amplitude........................7.1333 +- 0.5719
Background......................17.3032 +- 0.6111
----------------------------------------------------
Chi square....................................1.1068
----------------------------------------------------
```

and the fitting curve shown in fig.3 (left).

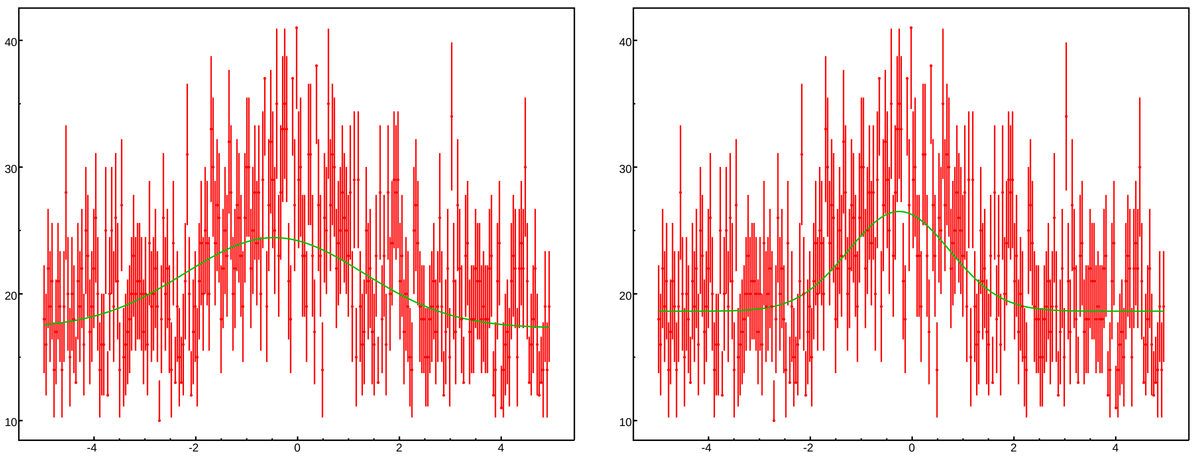

Fig. 3: Highly noisted data fitting example

This fit is not good enough, thus, we try

```
% fitter test.dat -o fit2.dat --std g -r
```

invoking robust fitting. This gets a better result:

```
----------------------------------------------------
Model....................................Gaussian
----------------------------------------------------
Mean...........................-0.2564 +- 0.1595
RMS.............................0.9985 +- 0.1638
Amplitude.......................7.8680 +- 1.2633
Background.....................18.6319 +- 0.5641
----------------------------------------------------
Chi square..................................1.0708
----------------------------------------------------
```

The fitting curve (fig.3, right) is much better this time.

5.2 SANS/YuMO example

To demonstrate the fitter usage, we have used the results of neutron measurements, obtained on the apoferritin protein sample obtained from Aldrich corporation. The file `apdn.dat` containing a SANS spectra is used as fitter input. A spherical shell model, which is one of the most adequate for apoferritin, is chosen. The corresponding command line is the following:

```
% fitter apdn.dat -o apdn.fit --yumo ss
```

This gives the following result:

```
----------------------------------------------------
Model................Spherical shell with resolution
----------------------------------------------------
Outer radius.....................59.6547 +- 0.0576
Inner radius.....................40.1314 +- 0.0700
Amplitude........................28.3414 +- 0.1721
Background........................0.0191 +- 0.0003
----------------------------------------------------
Chi square.................................31.2243
----------------------------------------------------
```

and the fitting curve shown in fig.4.

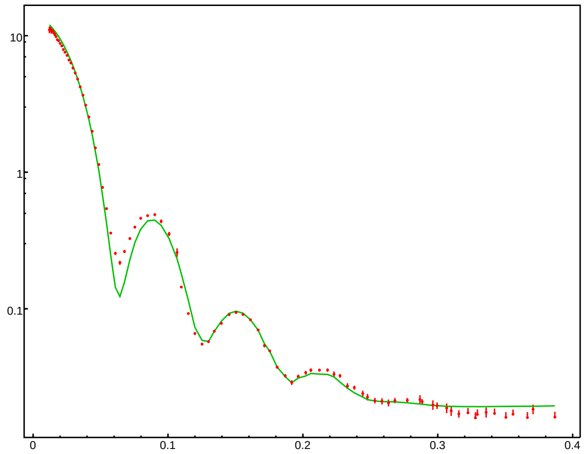

Fig. 4: Apoferritin data fitting example

6 Conclusion

Fitter is used as a part of a program complex on spectrometer YuMO. It is designed to be easily is understood by the experimenters. Both command line interface and GUI are provided. Despite of simplicity of offered models wide usage of fitter for many major appendices is supposed. The further development of the program is connected to expansion of offered models and introduction in the program of processing of the account of the structural factor. Fitter's design allows the simple adding of new models.

References

[1] *Soloviev A.G., Stadnik A.V., Islamov A.H. and Kuklin A.I.* FITTER. The package for fitting experimental data of the YuMO spectrometer by theoretical form-factors. Version 1.0. Long Write-Up and User's Guide. Communication of JINR **E10-2003-36**, Dubna: JINR, 2003.

[2] *D.J.Hughes, J.R.Wallace, R.W.Holzman* Phys.Rev.73, 1277 (1948).

[3] *Svergun, D. I.* (1993) A direct indirect method of small-angle scattering data treatment. J. Appl. Crystallogr. 26:258–267.

[4] *Svergun, D. I.* (1994) Solution scattering from biopolymers: advanced contrast variation data analysis. Acta Crystallogr. A. 50:391–402.

[5] *Svergun D. I.* (1992) Determination of the Regularization Parameter in Indirect-Transform Methods Using Perceptual Criteria. J. Appl. Crystallogr. 25, 495–503.

[6] *Svergun, D. I., C. Barberato, and M. H. J. Koch* (1995) CRYSOL: a program to evaluate x-ray solution scattering of biological macromolecules from atomic coordinates. J. Appl. Crystallogr. 28:768–773.

[7] *D. I. Svergun* (1999) Restoring Low Resolution Structure of Biological Macromolecules from Solution Scattering Using Simulated Annealing. Biophysical Journal Volume 76 June 2879–2886

[8] *Konarev, P. V., M. V. Petoukhov, and D. I. Svergun* (2001) MASSHA: a graphic system for rigid body modelling of macromolecular complexes against solution scattering data. J. Appl. Crystallogr. 34:527–532.

[9] *Kozin, M. B., and D. I. Svergun* (2000) A software system for automated and interactive rigid body modeling of solution scattering data. J. Appl.Crystallogr. 33:775–777.

[10] *A.I.Kuklin, A.Kh.Islamov, and V.I.Gordeliy* Two-detector System for Small-Angle Neutron Scattering Instrument, Neutron News, vol 16, 3, pp.16-18.

[11] *Yu.S. Kovalev, A.I. Kuklin, A.G. Novikov, V.V. Savostin, A.L. Shimkevich, E.L. Jadrovsky* The microstructure of Pb-K liquid alloy from small-angle neutron scattering experiments. Journal of Non-Crystalline Solids Volume 353, Issues 32-40, 2007, pp. 3532-3534.

[12] *T. N. Murugova, V. I. Gordeliy, A. I. Kuklin, I. M. Solodovnikova and L. S. Yaguzhinsky* Study of Three-Dimensionally Ordered Structures of Intact Mitochondria by Small-Angle Neutron Scattering. Crystallography Reports, Vol. 52, No. 3, 2007.pp.521-524.

[13] *Vyacheslav S. Molchanov, Olga E. Philippova, and Alexei R. Khokhlov, Yuri A. Kovalev and Alexander I. Kuklin* Self-Assembled Networks Highly Responsive to Hydrocarbons, Langmuir 2007, 23, 105-111

[14] *E.M.Kosacheva, D.B.Kudryavtsev, R.F.Bakeeva, A.I.Kuklin, A.Kh.Islamov, L.A.Kudryavtseva,V.F.Sopin and A.I.Konovalov* The aggregation of branched polyethylenimine and cationic surfactants in aqueous systems. Colloid Journal. Volume 68, Number 6 / December, 2006.

[15] *C.R.Haramagatti, A.Islamov, H.Gibhardt, N.Gorski, A.Kuklin and G.Eckold* Pressure induced phase transitions of TTAB-micellar solutions studied by SANS and Raman spectroscopy. Phys.Chem.Phys., 2006, 8, 994-1000.

[16] *A. Islamov, C.R. Haramagatti, H. Gibhardt, A. Kuklin, G. Eckold* Pressure-induced phase transitions in micellar solutions. Physica B 385–386 (2006) 791–794.

[17] *D.V. Lebedev, M.V. Filatov, A.I. Kuklin, A.Kh. Islamov, E. Kentzinger, R. Pantina, B.P. Toperverg, V.V. Isaev-Ivanov* Fractal nature of chromatin organization in interphase chicken erythrocyte nuclei: DNA structure exhibits biphasic fractal properties. FEBS Letters 579 (2005) 1465-1468.

[18] *A. S. Andreeva, A. I. Fomenkov, A. Kh. Islamov, A. I. Kuklin, O. E. Filippova and A. R. Khokhlov* Hydrophobic Aggregation in a Hydrophobized Polyacrylic Acid Gel Subjected to Microphase Separation. Polymer Science, Ser. A, Vol. 47, No. 2, 2005, pp. 194–201. Translated from Vysokomolekulyarnye Soedineniya, Ser. A, Vol. 47, No. 2, 2005, pp. 338–347.

[19] *A. N. Ozerin, D. I. Svergun, V. V. Volkov, A. I. Kuklin, V. I. Gordeliy, A. Kh. Islamov, L. A. Ozerina and D. S. Zavorotnyuk* The spatial structure of dendritic macromolecules, J. Appl. Cryst. (2005). 38, 996–1003.

[20] *Jaroslav Kriz, Josef Plestil, Herman Pospysil, Petr Kadlec, Cestmar Konak, Laszlo Almasy, and Alexander I. Kuklin* 1H NMR and Small-Angle Neutron Scattering Investigation of the Structure and Solubilization Behavior ofThree-Layer Nanoparticles. 11255 Langmuir 2004, 20, 11255-11263.

[21] *A. I. Kuklin, S. A. Kutuzov, A. Gabriel, G. Eckold, P. K. Utrobin, A. A.Smirnov, A. Kh. Islamov, A. S. Kirilov, A. A.Bogdzel, V. I. Gordeliy* The Parameters of Position Sensitive Detector with Central Hole and First Results in Small-Angle and Backscattering Configurations. Book of abstracts 4-th European Conference on Neutron Scattering, 25-29 June 2007, Lund, Sweden, p.530.

[22] *Feigin, L.A. & Svergun, D.I.* (1987) Structure analysis by small-angle X-ray and neutron scattering. New York: Plenum Press, 335 pp.

[23] *Yu.M.Ostanevich*, Time-of-flight small-angle scattering on pulsed neutron sources. Makromol. Chem., Macromol. Symp. **15**, 91-103 (1988)

[24] http://nfdfn.jinr.ru/ibr2.html

[25] http://wwwinfo.cern.ch/asdoc/minuit/minmain.html

[26] *P.Huber*, Robust statistics. Wiley, New-York (1981)

[27] *G.Ososkov*, Robust regression for the heavy contaminated sample. Proc. 2-nd International Tampere Conference in Statistics (1987) (Tampere, Finland)

[28] *N.Chernov*, *E.Kolganova*, *G.Ososkov*, Robust Methods for the RICH Ring Recognition and Particle Identification. Nuclear Inst. Meth. A433 (1999) 247-278.

[29] http://www.fltk.org

[30] http://www.jinr.ru/programs/jinrlib/gluplot/indexe.html