Software manual - SAXSFit

Brian R. Pauw
(Dated: December 1, 2010)

CONTENTS

GENERAL INTRODUCTION

This document attempts to explain the usage of some software packages written in Matlab revolving around the analysis of small-angle scattering patterns. The functions mostly take a few required input arguments, and optional arguments are supplied in so-called "parameter-value" pairs. This means that two arguments are supplied to the function, one "parameter", usually a string identifying the second argument, the "value". This looks like this in Matlab (where `>>` is the matlab prompt):

```
>>  [output_variable_1,output_variable_2,...,output_variable_n]=
        function_name(required_input_1,required_input_2,'parameter_1',1234,'parameter_2',4567);
```

These parameter-value pairs can be placed in any order (with one exception in "perfectpattern" programs), as long as the correct value follows the parameter. Limited input checks are done, but it is mostly up to the user to try to not make mistakes.

The code should be well-commented and readable, and no mistakes should remain. If, however, you do see room for improvement, please let me know so we can work on it!

SAXSFIT

purpose

This function is used to fit one-dimensional scattering curves to (classical) model functions. at its core lies a bound residual minimisation function written by John D'Errico, named "fminsearchbnd", which can be obtained from Matlab Central file exchange. The saxsfit function applies the minimisation function several times until no further improvement is obtained, which was found to be sometimes necessary when bad starting values are supplied. Furthermore, when a two-dimensional matrix of intensities is supplied, it is assumed that an evolving series of one-dimensional scattering patterns is given. These are fitted sequentially (the program should be able to figure out in which direction (rows or columns) the data lies), trying both the results from the previous fits as well as the initial guess as starting values. This may take a long time depending on the fitting model chosen, so an external loop is sometimes preferred to save time. Interpolation is implemented to save time, but no binning is implemented yet.

The least-squares minimisation function is defined as:

$$\epsilon^2 = \frac{\sum_{i=1}^{N} \left(\frac{I_{\mathrm{model,i}} - I_{\mathrm{measurement,i}}}{\sigma_{I,i}} \right)^2}{N - n_{\mathrm{param}}} \tag{1}$$

where i is the datapoint index, σ_I the datapoint uncertainty N is the total number of datapoints in the fit, and n_{param} is the number of fitting parameters.

usage

```
[qfit,Ifit,parameters,varargout]=saxsfit(q,I,funcname,initialguess,varargin)
```

q and I are the q and intensity values of the measured data, and are row- or column-vectors if only one dataset is to be measured or two-dimensional matrices for a series of datasets. Both q and I should be two-dimensional and of the same dimensions if a series is suplied. "funcname" is a string indicating the (basename of the) function to be fit, i.e. the filename of the matlab function without the ".m" extension. Some example functions are provided, which always have two inputs, q and a vector of fitting parameters, and always have one output, the calculated intensity. "initialguess" is a (row) initial guess vector with the same number of values as parameters in the model function.

The output parameters are $qfit$, the fitted q-values (which can be a subset of q when a "qlim" parameter is used), $Ifit$ the fitted intensities, and *parameters*, a vector or matrix with fitted values. An optional fourth output argument is the minimisation function output value (or vector of values in case of a series of datasets as input), so that multiple models can be compared.

Optional parameter-value pairs

examples

Fitting a single series with poisson statistics for the errors, indicating that the first and third parameter are the intensity scaling and background terms, respectively:

```
[qfit,Ifit,parameters,csqr]=saxsfit(q,I,...
   'Debye',[1 2 0],'separateintensity',[1 3],
   'Ierr',sqrt(I),'interpolationnumber',100,'qlims',[0.01 0.25],...
   'lowerlimit',[0 1 0],'upperlimit',[inf inf inf inf]);
```

The "upperlimit" here does not need to be strictly specified, since if absent, automatically assumes infinity.

Fitting a series of data merely requires q and I to be two-dimensional matrices:

```
[qfit,Ifit,parameters,csqr]=saxsfit(q,I,...
   'Debye',[1 2 0],'separateintensity',[1 3],
   'Ierr',sqrt(I),'interpolationnumber',100,'qlims',[0.01 0.25],...
   'lowerlimit',[0 1 0],'upperlimit',[inf inf inf inf]);
```

TABLE I. Parameter-value pairs and explanation.

Parameter	Value type	Description
lowerlimit	vector of same length as initial guess	lower constraint (bound) to fit parameters
upperlimit	vector of same length as initial guess	upper constraint (bound) to fit parameters
Ierr	vector of the same length as I	uncertainty values for the intensity
interpolationnumber	integer or fraction	interpolated intensity using interpolationnumber sampling points, or if fractional, the fraction of the number of datapoints is used
qlims	two-element floating point vector	lower and upper q bounds between which the fit should be attempted
separateintensity	two-integer vector	separate intensity (first integer) and background (second integer) optimisation. two-integer vector are indices of scaling and background parameters in model
reversecorrelation	zero (default) or one	if one, last measurement is fitted first, and the fitted parameters then are used as alternative initial guess for the n-1 measurement and so on

Usage for series without using the internal mechanism:

```
for ii=[198:400]; ...
   [qfit_pat,Ifit_pat,parameters_xul(ii,:),csqr_xul(ii)]=saxsfit(q(:,(ii)),I(:,(ii)),...
      'Pedersen',parameters_xul(ii-1,:),...
      'Ierr',0.01*(I(:,ii)),'interpolationnumber',100,'qlims',[0.004 0.30],...
      'lowerlimit',[0 0 0 0  1 0 4 0 0],...
      'upperlimit',[inf inf inf inf  4 inf 4 inf 0.1]);
   Ifit_xul(:,ii)=Ifit_pat;qfit_x(:,ii)=qfit_pat;
end
```

supplied fitting functions

Guinier

The Guinier function is a three-parameter fitting function, fitting a radius of gyration to the initial section of scattering pattern. Originally developed for monodisperse, isotropic, centrosymmetric and dilute samples at small angles. Conditions are that the sample consists of dilute, isotropic particles and that the q-range is limited to the initial section of the scattering pattern. In the case of polydispersity, the determined radius of gyration is the volume-squared weighted centre of mass:

$$R_{\text{weighted}} = \frac{\int V^2 P(r) r \mathrm{d}r}{\int V^2 P(r) \mathrm{d}r} \tag{2}$$

Where R, r can be the weighted radius of gyration and the particle radius of gyration, respectively, or (for other determinations) a weighted radius and particle radius. V is the volume of the particle, and $P(r)$ is the probability distribution function.

The parameters are:

- The intensity scaling factor
- The radius of gyration R_g
- A flat background

Debye

The Debye function is a three-parameter fitting function, fitting a correlation length l_c to the initial section of scattering pattern. Originally developed for describing the scattering pattern of random systems, it can be used to

describe a variety of structures. In case it is to be used for particular shapes, the correlation length can be related to the radius of gyration through $l_c^2 = \frac{1}{6}R_g^2$, although the range of applicability is slightly smaller than the Guinier function in this case. The function also converges to the Porod function at large ql_c.

The parameters are:

- The intensity scaling factor
- The correlation length l_c
- A flat background

USF

Beaucage's Universal Scattering Function has also been implemented, which is a combination of a Guinier term with a power-law slope. The power-law slope has a cut-off implemented by means of an error function. Its five parameters can describe a large range of systems, but the information from it is usually very limited in its usability. It can be applied to smoothly decreasing scattering patterns.

The parameters are:

- The Guinier term intensity scaling factor
- The radius of gyration R_g
- The Porod term intensity scaling factor
- The Porod term power law power (4 for a standard power-law slope).
- A flat background contribution.

Pedersen

This function is an seven-parameter function often used by prof. J.-S. Pedersen for fitting to a model of non-dilute polydisperse spheres. The model assumes a local monodisperse approximation for the fractal structure factor, and a Schultz distribution as size distribution. Should only be used if the data has been measured over a large q-range and only if the data does not show normal smoothly decreasing intensity behaviour (but shows bumps).

The parameters are:

- The intensity scaling factor
- The mean radius
- The variance in the radius
- The structure factor contribution scaling factor
- The Haussdorff fractal dimension (commonly $2 \leq D \leq 4$)
- A correlation length for the fractal region (causes a decrease in slope at low q)
- a background parameter

Pedersen_adapted

This function is a modification of the aforementioned "Pedersen" function, where the structure factor has been removed and replaced with a power-law contribution. This contribution can describe scattering originating from sources unrelated to the polydisperse particles, such as gels or very large particles or structures.

The parameters are:

- The intensity scaling factor of the power-law contribution

- The power of the power-law contribution (4 for a Porod-like slope)
- The intensity scaling factor of the polydisperse spheres contribution
- The mean of the polydisperse sphere radii (Schultz distribution)
- The variance of the polydisperse sphere radii (Schultz distribution)
- a background parameter

notes on your own implementations

With the supplied example functions, you should be able to modify some or write your own. A few things need to be taken into account here:

Firstly, the functions must always take two input parameters, the first being a q vector, and the second being a vector with model parameters. The output must always be the modeled intensity.

When implementing your own size distributions, make sure to check whether the volume-square weighting of the intensity contributions does not give significant contributions beyond the numerical integration limits for the parameter range of your choosing. This often means forcing an upper bound to the width of the distributions.