sas-temper: software for the analysis of small-angle scattering data[*]

William T. Heller[1], Mathieu Doucet[1], Richard K. Archibald[2]

[1]Neutron Scattering Division and [2]Computer Science and Mathematics Division; Oak Ridge National Laboratory; Oak Ridge, TN 37831.

1. Overview of *sas-temper*

The analysis of small-angle scattering (SAS) data can be time-consuming. It is also a challenge due to the diversity of models that can be used for fitting the data. Similarly, the fact that more than one model or set of parameters for a model can fit the measured data leaves many users uncertain about the quality of their results. These challenges give rise to a need to understand how reproducible the results of data fitting are and to automate that process to the greatest extent possible. The program *sas-temper* was developed to meet this need as part of an ORNL Laboratory Directed Research and Development project that investigated the use of artificial intelligence and machine learning for facilitating SAS data analysis.

sas-temper leverages the *sasmodels* package of the *SasView* project (https://www.sasview.org/) for both providing the set of models that can be used for data fitting and for many of the calculations involved in preparing a model intensity profile for comparison against the experimental data set being analyzed. Doing so allows users of sas-temper to directly leverage the excellent model documentation developed by the *SasView* team. Using the mechanisms provided in *sasmodels* and *SasView*, it is possible to employ custom models to *sas-temper* during data analysis. The fitting in *sas-temper* is performed using a variant of the traditional simulated annealing algorithm (Pincus, M., *Operation Research* **18**, 1225-1228 (1970)). The algorithm is described elsewhere. Plots generated by *sas-temper* are created using matplotlib (https://matplotlib.org/).

Unlike *SasView*, *sas-temper* runs from the command line, with the main driver being the ability to use computer clusters to run many jobs in parallel with minimal input and interaction from the user. The goal of *sas-temper* is to automate as much as possible to greatly reduce the demands on the time of people analyzing SAS data during the data fitting process.

sas-temper is open-source software and is available from

https://code.ornl.gov/wt3/sas_temper

[*] Notice: This manuscript has been authored by UT-Battelle, LLC, under contract DE-AC05-00OR22725 with the US Department of Energy (DOE). The US government retains and the publisher, by accepting the article for publication, acknowledges that the US government retains a nonexclusive, paid-up, irrevocable, worldwide license to publish or reproduce the published form of this manuscript, or allow others to do so, for US government purposes. DOE will provide public access to these results of federally sponsored research in accordance with the DOE Public Access Plan (http://energy.gov/downloads/doe-public-access-plan).

Installation instructions can be found at the link above. Documentation, examples, and tests can be downloaded from this location, as well.

A brief description of the required input for *sas-temper* and the results that are output follow.

Screenshots presented in this document of text were taken from files open in notepad++ (https://notepad-plus-plus.org/).

2. The Parameter File

An example of a *sas-temper* configuration file, which is a YAML file, is shown below. There are several options for configuring both how the program runs, as well as for configuring the specific model used for fitting the data. A full description of all options for configuring the model is presented below.

```
file:
   name: input_data.txt
   qmin: 0.01
   qmax: 0.30

output_files: cs_fit_sphere

sa_parameters:
   temperatures: 100
   temperature_rate: 0.90
   parameter_rate:  0.95
   iterations: 1000
   models_to_generate: 3

model:
   name: core_shell_sphere
   Category: shape-sphere
   scale:
      fixed: [1.0]
   background:
      fixed: [0.001]
   sld_core:
      linear: [-0.56, 8.00]
   sld_shell:
      linear: [-0.56, 8.00]
   sld_solvent:
      linear: [-0.56, 6.38]
   radius:
      log: [1.0, 3.0]
      polydispersity:
         SchulzDispersion: [0.0, 0.40]
   thickness:
      log: [0.0, 2.0]
      coupled: [radius]
      polydispersity:
         SchulzDispersion: [0.0, 0.40]
   Structure_Factor:
      name: hardsphere
      volfraction:
         linear: [0.0, 0.50]
```

The first block of the *sas-temper* configuration file specifies the data to be modeled as well as the q-range over which to perform the fitting.

```
file:
   name: input_data.txt
   qmin: 0.01
   qmax: 0.30
```

The data is assumed to be 3-, 4-, or 6-column ASCII text. Any metadata should be demarked as such with a non-numerical character at the start of the line. The program does not use any metadata present int he data files. The program will not accept data with other numbers of columns, or data in a different format. The columns, which are tab or space separated contain the following information. The 6-column format contains the data provided in the format that is output by the NIST Igor data reduction routines. The ORNL data reduction routines, *drt-sans*, produce data in the 4-column format.

column 1	**column 2**	**column 3**	**column 4**	**column 5**	**column 6**
Q	I	dI	dQ (σ_Q)	Q-bar	shadow factor

It is generally assumed that the units of *Q* are Angstroms^{-1}, while I is assumed to be provided in absolute units of cm^{-1}. Data that have not been scaled into absolute units can still be fit, but the scale factor cannot be fixed to 1.0. Data provided with *Q* specified in nm^{-1} can also be fit as long as length parameters are specified in nm and understood to be returned in the same. The units of the scattering length densities should be provided in units of 10^{-6} Angstroms^{-2} to ensure that the resulting models are in units of cm^{-1}.

The second block of parameters, which is the simplest of the set, specifies the name of the files that are output by the program.

```
output_files: fit_cs_sphere
```

All of the results that are output by *sas-temper* will be found in the directory in which the program is run and will be named “fit_cs_sphere*.*”. The results output are ASCII text of the model profile and png images that show the model fit to the data. The model profiles resulting from the fits are output as 2-column, tab separated ASCII text. Comments and other metadata in the output data are denoted by having the line in the file start with the character “#”. An analysis of the set of results is also output in text format, as are a set of plots presenting the various parameters found in the modeling. Both are described in Section 3.

In the third block of the configuration file, the parameters for configuring the simulated annealing engine used in *sas-temper* are specified.

```
sa_parameters:
   temperatures: 100
   temperature_rate: 0.90
   parameter_rate: 0.95
   iterations: 1000
```

```
models_to_generate: 3
```

The temperatures parameter specifies the number of times that the program will decrease the temperature in the simulated annealing process. Each time the temperature decreases, it is multiplied by the temperature_rate parameter, which must be a number 0.0 < temperature_rate < 1.0. The parameter_rate specifies how quickly the size of the change in the parameters can change during the simulated annealing process and must be a number 0.0 < parameter_rate < 1.0. This process is not part of a traditional simulated annealing algorithm. The iterations parameter specifies how many models are tested per temperature setting for a given fitting performed. In the example above, a total of 100,000 models will be tested during each of the five fittings performed (i.e. a total of 500,000 intensity profiles will be calculated) to produce a single fit to the data that is returned to the user as a result. The final parameter above, models_to_generate, specifies how many different results are returned to the user before *sas-temper* exits. The number has to be at least 1, while the upper limit is mostly dictated by the patience of the user. With a sufficiently large number (> 20), comparing the results from two different data sets becomes easier and can be subjected to statistical tests to provide a measure of whether the results truly differ or not.

The final block of parameters is specific to the model that the data will be fit with. A complete list of all models implemented in the *sasmodels* package and possible parameters are provided in a separate document (models_and_parameters.yaml) that can be found in the same directory of the code repository as this document. Rather than describing each parameter, the various keywords for the parameters are described below.

It is possible to employ a custom model by placing the source code (i.e. my_model.py) in the directory with the data and using the following in the configuration file.

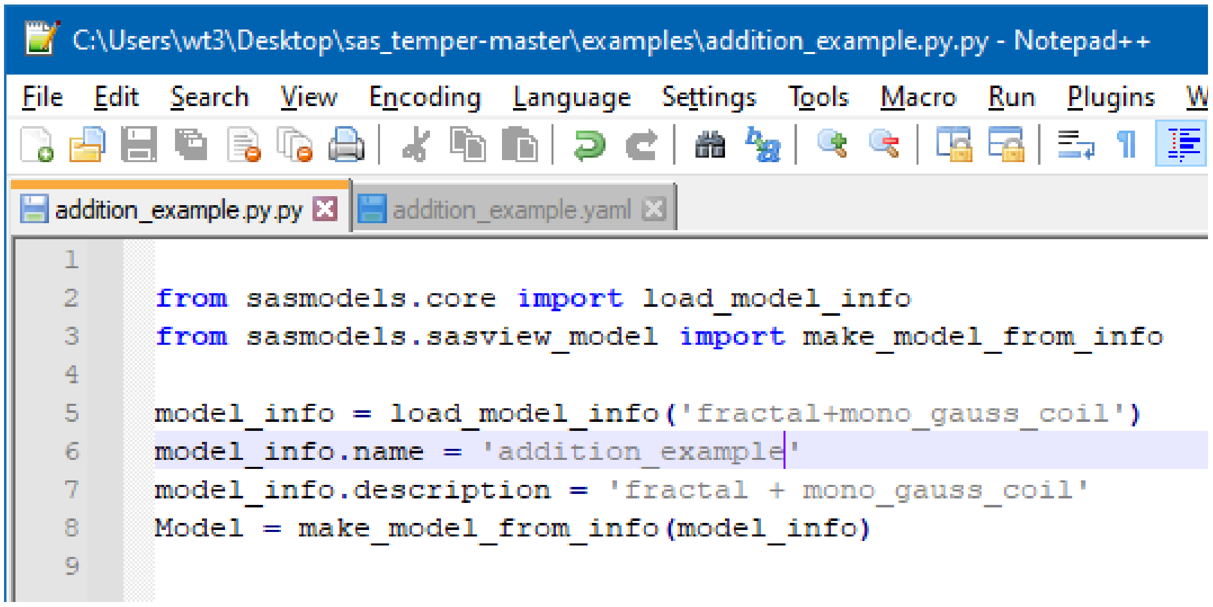
```
model:
  name: my_model.py
  Category: shape-independent
```

Custom models will also work in *SasView*, which provides a convenient way to debug the model. Models that are sums and products of existing *sasmodels* models can also be used. An example is shown in Figure 2.1. The easiest way to see what parameters need to be in the configuration file is by loading it into *SasView* as a plug-in model. The model shown in Figure 2.1 and the associated configuration file is provided in the repository. It is important to note that while the addition of two models works exactly as one would expect, the multiplication of two models, as one would do to apply a structure factor, works fine but does not appear to give access to the functionality related to the beta approximation for structure factors for working with non-spherical and polydisperse systems (Kotlarchyk, M. and Chen. S.-H., *J. Chem. Phys.* **79**, 2461-2469 (1983)). Users who require access to this functionality, which was implemented in *SasView* version 5 and later, should use *SasView* to analyze their data.

Parameters that are numbers are specified using four possible keywords: fixed, linear, log and integer.

fixed: [*value*]

The value of the parameter is always set to *value* throughout the modeling process.


C:\Users\wt3\Desktop\sas_temper-master\examples\addition_example.py.py - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins W
addition_example.py.py
addition_example.yaml

```

from sasmodels.core import load_model_info
from sasmodels.sasview_model import make_model_from_info

model_info = load_model_info('fractal+mono_gauss_coil')
model_info.name = 'addition_example'
model_info.description = 'fractal + mono_gauss_coil'
Model = make_model_from_info(model_info)

```

Figure 2.1. Example of a model that is a summation of two existing *sasmodels* models.

linear: [*min_value*, *max_value*]

The parameter is selected from a uniform distributrion from *min_value* ≤ parameter ≤ *max_value*. Using a linear range is best when the value is reasonably well known and can be constrained, such as for a scattering length density.

log: [*min_value*, *max_value*]

First, a value *X* is selected from a uniform distributrion from *min_value* ≤ *X* ≤ *max_value*. Then, the value of the parameter is set to 10^X. Using a logarithmic range for an initial fitting of data with an unknown structure is a good way to avoid spending a great deal of time engaging in trial-and-error searches for a good starting point.

integer: [*min_value*, *max_value*]

There are parameters within the sasmodels model set that use integers to pick from a set of options for the model profile calculation. Here, an integer value is selected from a uniform distribution within *min_value* ≤ parameter ≤ *max_value*.

There are other options that are modifiers on the parameters used for the calculations.

coupled: [*paramerer_name*]

While not intrinsic to *sasmodels* or *SasView*, *sas-temper* makes it possible to couple the value of one parameter to another parameter. Here, the value of the parameter is coupled to another that has the name *parameter_name*. When coupled, the value *V* selected for the parameter is used to calculate the value *Z* that is used in the model intensity profile calculation using parameter_name's value, denoted here as W. Specifically, *Z* = *VW*. When output with the results, the value shown is the actual value that is used for the intensity calculation.

polydispersity:
  Distribution_name: [*min_value*, *max_value*]

Many of the parameters of the models implemented in sasmodels can be made polydisperse. These parameters can also have a polydispersity applied to them in *sas-temper*. The built-in mechanisms for the calculations implemented in *sasmodels* are employed in *sas-temper*. Four options are available for the kind of polydispersity distribution employed when modeling with *sas-temper* that are specified as the *Distribution_name* parameter: *SchulzDistribution*, *GaussianDistribution*, *LogNormalDistribution* and *RectangleDistribution*. The width of the distribution, *w*, applied during the calculation is selected from a uniform distribution *min_value* ≤ *w* ≤ *max_value*.

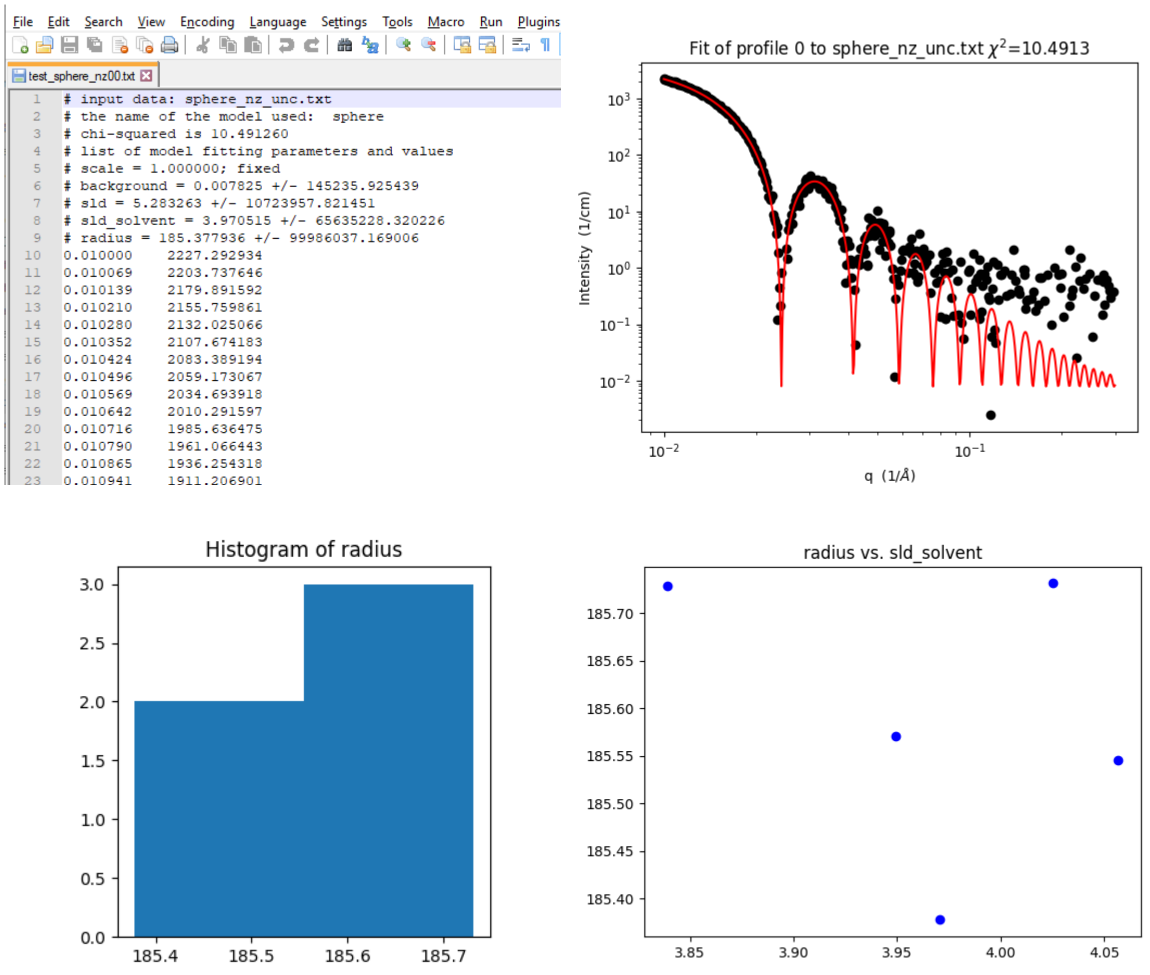
Structure_Factor:

The structure factors that are implemented in *sasmodels*, which can be found in the documentation at http://www.sasview.org/docs/user/qtgui/Perspectives/Fitting/models/index.html, are implemented in *sas-temper* as a model within a model. Parameters for the structure factors function exactly as they do for any parameter of a regular model.

It is important to note that *sas-temper* uses a very simple implementation of the structure factor that really makes it most appropriate to use for spherical particles that are monodisperse in size. The beta approximation (Kotlarchyk, M. and Chen. S.-H., *J. Chem. Phys.* **79**, 2461-2469 (1983)) for working with polydisperse and non-spherical particles is not accessible in *sas-temper* at present, and users should consider using *SasView* instead.

3. Results output by *sas-temper*

As was noted above, *sas-temper* outputs a variety of information that is useful when analyzing SAS data. The model intensity profiles found are output by the program as simple 2-column ASCII text files for easy plotting for inspection and publication. The error estimates in each model intensity profile are estimated from that particular result and are prone to seemingly odd values typical of such methods (as can be seen in the example in Figure 3.1). These uncertainties are not from the set analysis. Plots of the fitting results are also provided automatically by *sas-temper* in png format for easy inspection of individual results. Examples are shown in Figure 3.1. Importantly, plots of the sets of parameters found are also output by *sas-temper*, which are crucial for being able to evaluate the reproducibility of the fitting and any correlations that exist between parameters. Examples are also provided in Figure 3.1. These sets of plots

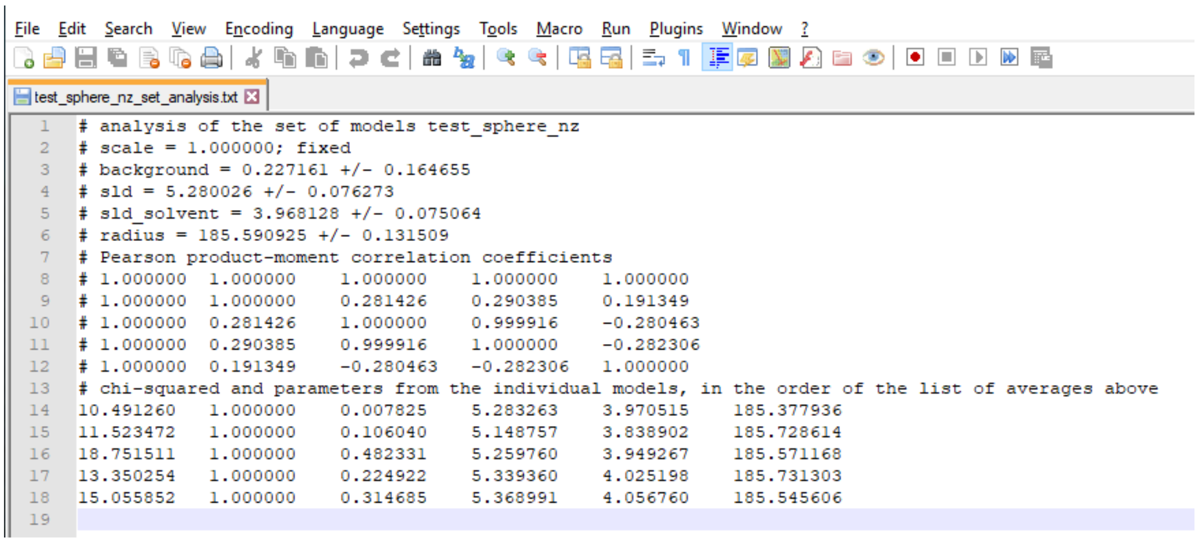


```
# input data: sphere_nz_unc.txt
# the name of the model used:  sphere
# chi-squared is 10.491260
# list of model fitting parameters and values
# scale = 1.000000; fixed
# background = 0.007825 +/- 145235.925439
# sld = 5.283263 +/- 10723957.821451
# sld_solvent = 3.970515 +/- 65635228.320226
# radius = 185.377936 +/- 99986037.169006
0.010000    2227.292934
0.010069    2203.737646
0.010139    2179.891592
0.010210    2155.759861
0.010280    2132.025066
0.010352    2107.674183
0.010424    2083.389194
0.010496    2059.173067
0.010569    2034.693918
0.010642    2010.291597
0.010716    1985.636475
0.010790    1961.066443
0.010865    1936.254318
0.010941    1911.206901
```

Figure 3.1. Top left: An example of the model intensity profile output after a single run of the fitting is complete. Top right: A plot of the model intensity profile (red line) shown in the top left panel with the data that was being fit (black circles). Bottom left: Example histogram of the radii found after all five independent runs of the fitting are finished. Bottom right: Example plot of two parameters used in the fitting (radius of the sphere and the SLD of the sphere) after all five independent runs of the fitting are finished that makes it easy to visualize correlations between the parameters in the fitting.

are useful when setting up additional runs of the fitting using more constrained ranges of possible parameter values and when investigating the possibility that more than one model may fit the data, which is a common problem in the analysis of SAS data.

Arguably, one of the most important files output by sas-temper is the final set of model parameters found and the analysis thereof. In addition to a table of the models and parameter values found during the fitting, the averages and standard deviations are output. The Pearson product-moment correlation coefficients between the various parameters are also provided. When combined with the histograms and the correlation plots, this summary file provides a powerful way to characterize both the reproducibility of the fit and is extremely useful for developing better constraints to use when trying to refine the analysis of a specific data set. An example of this set analysis file is presented in Figure 3.2. The file is provided as ASCII text, and columns of information are tab separated for easy import or copy-paste into a spreadsheet or table.


File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
test_sphere_nz_set_analysis.txt

```
# analysis of the set of models test_sphere_nz
# scale = 1.000000; fixed
# background = 0.227161 +/- 0.164655
# sld = 5.280026 +/- 0.076273
# sld_solvent = 3.968128 +/- 0.075064
# radius = 185.590925 +/- 0.131509
# Pearson product-moment correlation coefficients
# 1.000000  1.000000    1.000000    1.000000    1.000000
# 1.000000  1.000000    0.281426    0.290385    0.191349
# 1.000000  0.281426    1.000000    0.999916    -0.280463
# 1.000000  0.290385    0.999916    1.000000    -0.282306
# 1.000000  0.191349    -0.280463   -0.282306   1.000000
# chi-squared and parameters from the individual models, in the order of the list of averages above
10.491260   1.000000    0.007825    5.283263    3.970515    185.377936
11.523472   1.000000    0.106040    5.148757    3.838902    185.728614
18.751511   1.000000    0.482331    5.259760    3.949267    185.571168
13.350254   1.000000    0.224922    5.339360    4.025198    185.731303
15.055852   1.000000    0.314685    5.368991    4.056760    185.545606
```

Figure 3.2. An example of the set analysis file produced after a complete run of *sas-temper*.