Two practical Java software tools for small-angle X-ray scattering analysis of biomolecules

Andreas Hofmann\textsuperscript{1,2,}\textsuperscript{*} and Andrew Whitten\textsuperscript{3,}\textsuperscript{*}

\textsuperscript{1}Structural Chemistry Program, Eskitis Institute, Griffith University, Nathan, Queensland, Australia
\textsuperscript{2}Faculty of Veterinary Science, The University of Melbourne, Parkville, Victoria, Australia
\textsuperscript{3}Institute for Molecular Biosciences, University of Queensland, St Lucia, Queensland, Australia

*Corresponding authors: \texttt{a.hofmann@griffith.edu.au}
\texttt{a.whitten@uq.edu.au}

Word count: 4389
Character count: 27954
Last modified: 13 Mar 2014

Keywords: molecular modelling, rigid-body fitting, small-angle X-ray scattering
Abstract
Small-angle X-ray scattering has established itself as a common technique in structural biology research. Here, we describe two novel Java applications to aid modelling of three-dimensional macromolecular structures based on small-angle scattering data. 

MolScat is an application that computes small-angle scattering intensities from user-provided three-dimensional models. The program can fit the theoretical scattering intensities to experimental X-ray scattering data. SAFIR is a program for interactive rigid body modelling into low resolution shapes restored from small-angle scattering data. The program has been designed with an emphasis on ease of use and intuitive handling. An embedded version of MolScat is used to enable quick evaluation of fit between model and experimental scattering data. SAFIR also provides options to refine macromolecular complexes with optional user-specified restraints against scattering data by means of a Monte-Carlo approach.

1. Introduction
Small-angle X-ray scattering (SAXS) has become a regularly used technique in the past decade to characterize the structure of biological macromolecules. The fact that protein samples originally being prepared for NMR or X-ray crystallographic studies are also appropriate to be subjected to SAXS experiments is one major reason for the popularity of this technique. At the same time, synchrotron facilities are increasingly establishing SAXS beam lines dedicated to structural biology thus making the technique even more accessible.

A frequent experimental question to be addressed in structural biology is the characterisation of the quaternary structure of proteins and their complexes in solution. A typical work flow for modelling such structures comprises rigid body fitting of atomic models of individual monomers. This task is greatly aided by ab initio calculation of three-dimensional shapes of the scattering object, typically represented as accumulations of dummy atoms, beads or density maps (Chacón et al., 1998; Svergun, 1999; Svergun et al., 2001; Walther et al., 2000). Using these ab initio shapes as guides, the individual models can in many cases be manually arranged to approximately fill the restored volume. To further refine such an approximate model, small re-positioning and adjustments of the relative orientation of the individual monomers are required and the fit of the model-derived scattering intensity to the experimental data needs to be evaluated. Model adjustments can be made either manually or carried out computationally.

Here, we describe two practical Java software programs that can perform these tasks. SAFIR is an application that allows for visual arrangement of protein monomers into quaternary structures. From the SAFIR application, the theoretical scattering intensity of a model can be computed and its fit to experimental data be evaluated. This latter task is carried out by the Java software MolScat which can also be used as a standalone program.

Different approaches and software tools have been developed to generate scattering intensity profiles from molecular models. CRYSOL is the most popular software in this context and uses multipole expansion to calculate a scattering profile based on atomic coordinates (Svergun et al., 1995). In contrast, the FoXS server uses the Debye formula to calculate scattering profiles (Schneidman-Duhovny et al., 2010), and ORNL_SAS performs a Monte-Carlo sampling of the inter-atomic distances in the model (Tjioe & Heller, 2007). Yet a different method to speed up calculation applies coarse graining by combining several atomic scatterers into a scattering unit (Grishaev et al., 2005; Wriggers, 2010; Yang et al., 2009).

In order to perform rigid body modelling of multimeric complexes, any molecular graphics software can be used and the generated model, output in PDB format, be subjected to evaluation of its small-angle scattering intensity using the software programs above. Many rigid body modelling programs


in this context focus on the computational modelling/docking aspects and rely on external graphics software for visualisation and manual manipulation. Situs, for example, recommends the molecular graphics program VMD (Humphrey et al., 1996) for this purpose (http://situs.biomachina.org/tutorial_saxs.html). Similarly, sas_rigid (Meesters et al., 2010) is centred around the Monte-Carlo computations, outsourcing the scattering intensity evaluation to CRYSOL or CRYSON (Svergun et al., 1998). The program MASSHA offers model visualisation and manipulation, but also has a built-in feature to compute the scattering intensity of the model (Konarev et al., 2001).

Within the ongoing project of developing fundamental Java classes and applications for structural biology and biophysical chemistry research (Hofmann & Wlodawer, 2002), we set out to design SAFIR, a simple-to-use and portable Java application that aids in modelling of quaternary protein structures using solution scattering data. During this process, it became apparent that the calculation of theoretical scattering intensities from atomic models would also need to be implemented, and we therefore developed the standalone application MolScat.

2. Program implementation and methods

2.1 MolScat: General concept

MolScat is a program to evaluate solution scattering of biological macromolecules from atomic coordinates. It can be run with terminal commands given when starting the program, or through a graphical user interface. The program reads three-dimensional structures provided as PDB files and considers non-water atoms to calculate a scattering intensity curve. If experimental scattering data are provided, MolScat will fit the theoretical scattering curve to the experimental data. Results are provided in form of graphical plots, an ASCII file of the theoretical scattering data (and fit with goodness of fit if applicable), as well as selected biophysical parameters.

After determining the bounding box of the model, this box is divided into a voxel grid with each voxel having a side length of 3 Å. In the grid, voxels representing the inside, surface and envelope of the protein are identified. The envelope is the first shell of unoccupied voxels around the model. From the voxel grid, the pair distance distribution function $p(r)$ for the protein is generated as a histogram and smoothed using the KernelEstimator class from the WEKA package (Hall et al., 2009).

2.2 MolScat

To evaluate X-ray scattering, the pair distance distribution function $p(r)$ for the provided three-dimensional atomic model is generated by evaluating the electron density in each voxel (number of electrons in each voxel divided by the voxel volume $0.027 \text{ nm}^3$). Water molecules in the model are automatically removed. To enable explicit corrections for excluded volume and the solvation shell contrast, we have implemented a rigorous calculation of the pair distance distribution. This leads to six components of the pair distribution function, describing the convolutions between macromolecule and envelope as well as solvent voxels:

\[
\begin{align*}
  p_1(r) &= 2 \sum_{i,\text{prot}} \sum_{j,\text{prot} > i,\text{prot}} \rho(r_{i,\text{prot}}) \rho(r_{j,\text{prot}}) \\
  p_2(r) &= -2 \rho_s \left[ \sum_{i,\text{prot}} \sum_{j,\text{prot} > i,\text{prot}} \rho(r_{i,\text{prot}}) + \rho(r_{j,\text{prot}}) \right] + \sum_{i,\text{prot}} \sum_{j,\text{env}} \rho(r_{i,\text{prot}}) \rho(r_{j,\text{env}}) \\
  p_3(r) &= 2 \sum_{i,\text{prot}} \sum_{j,\text{env}} \rho(r_{i,\text{prot}}) \rho(r_{j,\text{env}}) \\
  p_4(r) &= -2 \rho_s \left[ \sum_{i,\text{env}} \sum_{j,\text{env} > i,\text{env}} \rho(r_{i,\text{env}}) + \rho(r_{j,\text{env}}) \right] + \sum_{i,\text{prot}} \sum_{j,\text{env}} \rho(r_{i,\text{env}}) \rho(r_{j,\text{env}}) \\
  p_5(r) &= 2 \sum_{i,\text{env}} \sum_{j,\text{env} > i,\text{env}} \rho(r_{i,\text{env}}) \rho(r_{j,\text{env}})
\end{align*}
\]
Here, the subscripts $i$ and $j$ refer to voxels belonging to either the macromolecule (prot) or the solvent envelope (env). Hence, $\rho(r_{prot})$ refers to the electron density of a voxel containing atoms of the macromolecule, while $\rho(r_{env})$, refers to the electron density of the solvent envelope. The electron density of the bulk solvent is represented by $\rho_s$ (default: 334 e nm$^{-3}$).

The theoretical scattering intensity $I_t(q)$ is calculated by

$$I_t(q) = 4\pi \int p(r) \left[ \sin(q \cdot r) / (q \cdot r) \right] dr$$

for each of the six pair distance distribution functions. The total scattering intensity can be calculated from the following equation that allows optimisation of two scaling factors, one for the excluded volume $F_{prot}$ and one for the solvation shell contrast $F_{env}$:

$$I(q) = F_{prot}^2 I_1(q) + F_{prot} F_{env} I_2(q) + F_{env} I_3(q) + F_{env}^2 I_4(q) + I_5(q)$$

The two scaling factors are determined by optimising the fit of the total scattering intensity $I(q)$ to the experimental data $I_e(q)$. The scaling of theoretical to experimental scattering intensities, as well as an intensity background is calculated by linear regression

$$I_e(q) = \text{scale} \cdot I_t(q) + \text{background}$$

The goodness of fit between the two data sets is evaluated using the $\chi$ value as defined by Svergun (Svergun et al., 1995)

$$\chi = \left\{ \frac{1}{N-1} \sum \left[ \left( I_e(q) - \text{scale} \cdot I_t(q) - \text{background} \right) / \sigma_e \right]^2 \right\}^{1/2}$$

where $N$ is the number of experimental data points included in the fit. The final scattering intensity data are then calculated by spline interpolation to yield data points for each angular momentum tabulated in the user-provided experimental data file.

### 2.3 SAFIR

SAFIR (Small-angle scattering data Fitting with Rigid Bodies) is an application to fit small-angle scattering data with rigid body objects, with the main purpose of modelling oligomeric structures of biological macromolecules. In the design of the program, a clear emphasis has been intuitive and ease of usage. Three-dimensional atomic protein models are therefore rendered as Cα traces and individual monomers are automatically coloured differently. Multimeric models can be established by loading individual monomers or by loading a PDB file with monomers being recognised by their chain identifier.

The program allows loading of individual protein monomers or oligomeric structures which can be displayed and oriented as rigid bodies in the embedded Jmol molecular graphics viewer (http://jmol.sourceforge.net/). A shape object restored from small-angle scattering data can also be displayed, enabling the user to arrange the protein molecules to fit. For the loaded model, the small-angle scattering can be evaluated and compared to experimental scattering data using an embedded version of MolScat. Using molecular viewer features inherited from Jmol, the screen representation of structures can be adjusted using the mouse. Modification of position and orientation of one or
more individual monomers is achieved by pressing NumPad keys.

Other molecular graphics features included in SAFIR comprise an alignment tool, as well as clash
check and visualisation. The structural alignment algorithm is based on inertia axes, thus allowing
for superposition of high and low resolution models. The algorithm follows the concept by Svergun
and colleagues (Kozin & Svergun, 2001) which is based on a distance measure first introduced for
polyhedra matching (Bloch et al., 1993).

Rigid body refinement of the loaded model has been implemented by means of a Monte-Carlo
approach which uses the fit between model-derived and experimental scattering data as target
function. The implemented protocol runs through the following steps:

1. Initial values for $\chi$ and $R_g (R_{g\text{Start}})$ are calculated for the starting model.
2. A random movement of all rigid bodies activated by the user is achieved by a translation
   vector and three rotations (one each around the x-, y-, and z-axis). For this purpose, six
   positive random numbers are generated per rigid body to make up the translation vector
   and the three rotations. Another six random numbers per rigid body determine whether any
   of the components should be positive or negative. One further random number is required that
   provides a seed for the random number generation in the next cycle.
3. The theoretical scattering of the new model is evaluated using MolScat. The current $R_g$
   is stored as $R_{g\text{Now}}$. If the user specified distance restraints, these are evaluated and a penalty
   of 0.1 is applied to the MolScat-derived $\chi$ value per violated restraint.
4. If the $\chi$ value improved as compared to the previous cycle, a clash analysis is performed on
   the current model if request by the user. If more than the tolerated number of clashes are
   observed, the new model is discarded and the step counted as unproductive (“dead cycle”).
   Otherwise the model is kept and subjected to a new cycle. If the $\chi$ value did not improve
   compared to the previous cycle, the move is accepted with a probability that is proportional
   to $e^{-\frac{d}{T}}$ where $T$ is the current 'temperature' of the model (see below).
5. If the number of unproductive cycles exceeds the user-set value of dead cycles, the shift
   sizes are decreased to the new value of $\text{coolingFactor} \cdot \text{oldValue}$. At the same time, the
   number of dead cycles is increased to 1.2 times the current value. If the miminum shift size
   has been reached or the number of unproductive cycles has reached the maximum number of
   final cycles specified by the user, the procedure will exit. Otherwise, a new cycle is started
   at step (2).

The algorithm accepts a bad move ($\chi_{\text{now}} \geq \chi_{\text{previous}}$) in step (4) above with a probability

$$p = \text{toleranceFactor} \cdot e^{-\frac{d}{T}}.$$ 

The $\text{toleranceFactor}$ is a user-provided variable (default: 0.4), and $d$ is calculated as

$$d = \chi_{\text{now}} - \chi_{\text{previous}} + | R_{g\text{Now}} - R_{g\text{Start}} |.$$ 

### 2.5 Availability

Both programs make use of and extend Java classes previously developed in our laboratory
(Hofmann & Wlodawer, 2002; Weeratunga et al., 2012). They are available as standalone compiled
Java applications from the project home page at http://www.structuralchemistry.org/pcs/. The
MolScat API includes methods that enable interfacing with other Java applications and may thus
also be useful to developers. The applications are freely available to academic users. For download,
users will be asked for their name, institution and email address. The source code is available from the authors upon request.

3. Results
3.1 Specific consequences of the voxel concept
As a consequence of the allocation of electron density into voxels, the radius of gyration $R_g$ of the non-solvated (dry) model of lysozyme is larger than that of the solvated model (Table 1). Due to binning of electrons into voxels of $27\text{Å}^3$ volume, the centre of mass of the binned electron density may be further from the centre of the macromolecule than the actual atom the electrons belong to. This effect may be of particular importance at the periphery of a macromolecule where there is a lower packing density of atoms, and its impact on the overall electron density distribution will be more pronounced in smaller molecules. Accordingly, a comparatively larger $R_g$ value is observed for the non-solvated model.

Clearly, by considering the excluded volume ($F_{prot}$) and the envelope contrast ($F_{env}$), this effect is corrected for, and very sensible $R_g$ values for the solvated models are obtained. In the case of lysozyme, the factor correcting for the excluded volume is close to 1, and, concomitantly, the correction factor for the electron density of the solvent layer is reduced to counteract these effects. Conceptually, there is thus a caveat in the interpretation of $F_{prot}$ and $F_{env}$. Although the definition of these parameters is clear, we feel that these factors also help to correct for inadequacies in the modelling procedure.

3.2 Benchmarking of MolScat
Four protein systems with published small-angle scattering data have been used to compare the performance of MolScat to that of other generally available programs. The results are listed in Table 1, and demonstrate that MolScat computes scattering intensities from protein models similar to those obtained with other software. Importantly, the quality of fit between the computed and experimental scattering intensities is highly similar for all algorithms. Among the tested algorithms, only Crysol is a standalone program and therefore provides the only comparison for computing time. The four examples show that MolScat calculations are slower than those of Crysol, with an exponential increase in computing time as the size of protein system increases. The longer computing time required by MolScat may partly result from the methodology chosen here. Furthermore, we have placed an emphasis on the conceptual design, and some improvements may be possible when optimising the source code for speed. However, a substantial contribution to computing time arises from the Java programming language itself which is known to be less time efficient as compared to languages such as Fortran and C (Amedro et al., 2008; Ashby, 2003).

3.3 Rigid body fitting with SAFIR
For illustration, we have re-worked the rigid body fitting of the VILIP-1 homodimer published previously (Wang et al., 2011). VILIP-1 belongs to the family of neuronal calcium sensor (NCS) proteins, and GASBOR (Svergun et al., 2001) was used for ab initio shape restoration from SAXS data obtained with VILIP-1 in the presence of calcium and reducing conditions. The shape obtained was distinctly different to the dimer models of other NCS proteins, suggesting that although these proteins share a similar overall fold, they may have different molecular mechanisms. The VILIP-1 dimer model proposed in a previous modelling study (Li et al., 2011) was used as initial model and superimposed on the GASBOR shape in SAFIR (Figure 2A) using the in-built superposition algorithm based on inertia axes. The model shows reasonable agreement with the small-angle X-ray scattering data collected at 12 mg mL$^{-1}$ protein concentration (Figure 2B). This fit was evaluated by
using the *MolScat* implementation in *SAFIR* using two mouse clicks. Manual adjustments of the model yielded varying changes of the goodness-of-fit parameter $\chi^2$, but no significant overall improvement. Thus, the model was subjected to computational rigid-body fitting by the Monte-Carlo algorithm outlined above (Figure 2C). Within five minutes of computation time, a dimer conformation close to the proposed model but with significantly improved fit to the small-angle X-ray scattering data was obtained (Figure 2D). Distance restraints can be added before starting the computational refinement, but have not been used in this example.

4. Conclusions
With *SAFIR*, we present a novel interactive modelling software that is tailored for rigid body modelling applications and low resolution structural data from small-angle X-ray scattering. Similar to **MASSHA** (Konarev *et al.*, 2001), our software combines several features of low resolution rigid body modelling with SAXS data into one application, and we have put a strong emphasis on intuitive use. Through the built-in molecular graphics capability provided by *Jmol*, model handling and inspection is highly interactive and intuitive and builds on the experiences users have accumulated from other common molecular graphics software such as *O* (Jones *et al.*, 1991) and *COOT* (Emsley & Cowton, 2004). Manipulation of models is accessibly and conveniently done by arrow keys on the number pad, omitting intermittent steps to change from viewing to transformation mode. The fit of the present model to the experimental data can be evaluated with a mouse click. Computational refinement of a model is possible in the current version of the software by means of a Monte-Carlo procedure, which is a commonly chosen method for this type of refinement (Meesters *et al.*, 2010).

Within the concept of our Java project **PCSB** (Hofmann & Wlodawer, 2002), it is one of our main goals to have an intuitive, easy-to-use and portable application. We therefore had to implement an algorithm to compute small-angle X-ray scattering intensities from molecular models. This application, *MolScat*, is part of the *SAFIR* modelling program, but can also be used on its own. The results generated by *MolScat* are in agreement with other available software; the main difference at this stage is the longer computation time. For the frequently used one-off calculations, we do not consider the longer computation time a significant problem, since even larger protein systems are still processed well under 1 min. However, time is certainly a more significant aspect in repeated executions of *MolScat*, such as for example in the Monte-Carlo refinement, a feature also implemented in *SAFIR*. Given the computing power of modern CPUs and the fact that the automated refinement is carried out only for improvement of the pre-oriented model provided by the user, we do not feel that the computing time has a major impact on the benefit of this software for the user. However, in future versions of this software will address this issue and work towards improvement of the calculation speeds, e.g. by implementing Lookup Table optimisation (Wilcox *et al.*, 2011).

Additionally, future work on these programs will include implementation of neutron scattering in *MolScat*, as well as more specialised molecular modelling options in *SAFIR*, such as symmetry restraints, mixtures of oligomeric species and generation and handling of different conformations.

**Author contributions**
AH and AW invented and designed algorithms, tested the programs and wrote the manuscript. AH wrote the software and manuals.
**Acknowledgements**

Research in the lab of AH is funded by the National Health and Medical Research Council (1002898, 1044022) and the Australian Research Council (LE120100071, DP140100599). AEW was supported by an NHMRC Peter Doherty Fellowship (569864). Parts of this research were undertaken on the SAXS/WAXS beamline at the Australian Synchrotron, Victoria, Australia. 

*Conflict of interest:* none declared.
References
### Tables

**Table 1**
Comparison of results obtained with different programs for fitting SAXS data

<table>
<thead>
<tr>
<th></th>
<th>MolScat</th>
<th>CRYSOL</th>
<th>FoXS</th>
<th>Sastbx</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Example 1: Lysozyme</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Computation time (s)</td>
<td>1.6</td>
<td>0.45</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Goodness of fit $\chi$</td>
<td>0.52</td>
<td>0.45</td>
<td>0.45</td>
<td>0.45</td>
</tr>
<tr>
<td>$R_g$ of dry model (Å)</td>
<td>16.3</td>
<td>14.0</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>$R_g$ of solvated model (Å)</td>
<td>15.4</td>
<td>15.0</td>
<td>14.0</td>
<td>-</td>
</tr>
<tr>
<td>Volume of dry model (Å$^3$)</td>
<td>13716</td>
<td>17350</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Volume of solvated model (Å$^3$)</td>
<td>13063</td>
<td>17410</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Contrast factors</td>
<td>1.05, 0.20</td>
<td>-</td>
<td>1.01, 0.59</td>
<td>-</td>
</tr>
<tr>
<td><strong>Example 2: VILIP-1 dimer</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Computation time (s)</td>
<td>7.9</td>
<td>0.61</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Goodness of fit $\chi$</td>
<td>5.2</td>
<td>4.3</td>
<td>3.0</td>
<td>4.4</td>
</tr>
<tr>
<td>$R_g$ of dry model (Å)</td>
<td>29.6</td>
<td>27.6</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>$R_g$ of solvated model (Å)</td>
<td>29.8</td>
<td>28.7</td>
<td>28.2</td>
<td>-</td>
</tr>
<tr>
<td>Volume of dry model (Å$^3$)</td>
<td>40500</td>
<td>52000</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Volume of solvated model (Å$^3$)</td>
<td>67500</td>
<td>53190</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Contrast factors</td>
<td>0.60, 1.04</td>
<td>-</td>
<td>1.04, 3.09</td>
<td>-</td>
</tr>
<tr>
<td><strong>Example 3: 14-3-3β dimer</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Computation time (s)</td>
<td>17</td>
<td>0.74</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Goodness of fit $\chi$</td>
<td>1.9</td>
<td>1.4</td>
<td>2.0</td>
<td>1.0</td>
</tr>
<tr>
<td>$R_g$ of dry model (Å)</td>
<td>30.0</td>
<td>28.2</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>$R_g$ of solvated model (Å)</td>
<td>30.2</td>
<td>30.4</td>
<td>28.2</td>
<td>-</td>
</tr>
<tr>
<td>Volume of dry model (Å$^3$)</td>
<td>50463</td>
<td>63090</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Volume of solvated model (Å$^3$)</td>
<td>48060</td>
<td>64760</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Contrast factors</td>
<td>1.05, 1.62</td>
<td>-</td>
<td>1.05, 4.00</td>
<td>-</td>
</tr>
<tr>
<td><strong>Example 4: Glucose isomerase</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Computation time (s)</td>
<td>62</td>
<td>1.0</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Goodness of fit $\chi$</td>
<td>0.6</td>
<td>0.35</td>
<td>0.31</td>
<td>0.38</td>
</tr>
<tr>
<td>$R_g$ of dry model (Å)</td>
<td>34.2</td>
<td>31.8</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>$R_g$ of solvated model (Å)</td>
<td>33.0</td>
<td>33.5</td>
<td>31.7</td>
<td>-</td>
</tr>
<tr>
<td>Volume of dry model (Å$^3$)</td>
<td>170019</td>
<td>213100</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Volume of solvated model (Å$^3$)</td>
<td>161922</td>
<td>216800</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Contrast factors</td>
<td>1.05, 0.2</td>
<td>-</td>
<td>1.05, 0.16</td>
<td>-</td>
</tr>
</tbody>
</table>
References for model and data used in above examples: example 1 (Svergun et al., 1995), example 2 (Wang et al., 2011), example 3 (Hu et al., 2012), example 4 (Whitten, unpublished). All calculations were carried out on a Linux PC (Intel i7-2620M QuadCore, 7.7 GB RAM; Fedora Core 16.x86_64) and considered the solvation model provided by each software. CRYSOL (Svergun et al., 1995) obtained from http://www.embl-hamburg.de/biosaxs/crysol.html; FoXS at http://modbase.compbio.ucsf.edu/foxs/; Sastbx at http://sastbx.als.lbl.gov/cgi-bin/index.html.
Figure legends

Figure 1
Screenshot of a MolScat calculation fitting lysozyme small-angle X-ray scattering data provided with CRYSOL (Svergun et al., 1995). The top panel shows the graphical user interface and the parameters used for the calculation. Alternatively, the program can be invoked with terminal commands. The medium panel shows a plot of the fit of the model to the experimental data. This window can be suppressed by the user if invoking the program from the terminal. In the bottom panel, the pair distance distribution function of the model, calculated by the algorithm outlined in the text, is plotted. The graphs can be saved as binary images or ASCII data.

Figure 2
Screenshots from a rigid-body fitting task with SAFIR. A The dimer model was loaded at once and individual molecules are automatically recognised from their chain identifiers and coloured differently. Red colour indicates clashes identified by the clash check (Check Model menu). Molecules can be transformed (translation/rotation) separately or in groups as per the user's choice (Molecules button in the tool bar). The ab initio shape is shown as magenta sphere. B Experimental SAXS data are shown as red dots and the fit of the current model on the Molecular Viewer panel (blue line) can be evaluated by the embedded version of MolScat. C Computational rigid body fitting was carried out using the input provided on MC Ref panel. All parameter fields are populated with default values and can be changed by the user. D View of the final model obtained after 31 steps of Monte-Carlo refinement using the parameters shown in C.