



JOURNAL OF
SYNCHROTRON
RADIATION

New data analysis for BioSAXS at the ESRF

Jérôme Kieffer, Martha Brennich, Jean-Baptiste Florial, Marcus Oscarsson, Alejandro De Maria Antolinos, Mark Tully and Petra Pernot

CONFIDENTIAL – NOT TO BE REPRODUCED, QUOTED NOR SHOWN TO OTHERS

SCIENTIFIC MANUSCRIPT

For review only.

Wednesday 04 May 2022

Category: *computer programs*

Co-editor:

Telephone:

Fax:

Email:

Submitting author:

Jérôme Kieffer

ADA, ESRF, 71 avenue des Martyrs, Grenoble, 38000, France

Telephone: +33476882445

Fax: +33476882020

Email: jerome.kieffer@esrf.fr

New data analysis for BioSAXS at the ESRF

JÉRÔME KIEFFER,^{a*} MARTHA BRENNICH,^b JEAN-BAPTISTE FLORIAL,^b
MARCUS OSCARSSON,^a ALEJANDRO DE MARIA ANTOLINOS,^a MARK TULLY^a AND
PETRA PERNOT^a

^a*The European Synchrotron, 71 Avenue des Martyrs, 38000 Grenoble France, and*

^b*European Molecular Biology Laboratory, 71 Avenue des Martyrs, 38000 Grenoble
France. E-mail: jerome.kieffer@esrf.fr*

**BioSAXS; online data analysis; solution scattering; proteins; biological small-angle X-ray scattering;
automation; high brilliance; structural biology; high-throughput SAXS; size-exclusion chromatography; online
purification**

Abstract

The second phase of the ESRF upgrade program did not only provide a new storage ring (EBS), it also allowed to refurbish several beam-lines, including the BioSAXS BM29, which got a larger detector in a new flight tube. To cope with the resulting increased data-flux, analysis software needed to be rewritten to continue to ensure real-time processing. This article describes *FreeSAS*, an open-source collection of various SAXS analysis algorithms needed to reduce BioSAXS data, and *dahu*, the tool used to interface data-analysis with beam-line control. It further presents the data processing pipelines for the different data acquisitions modes of the beam-line, using either a sample-changer for individual homogeneous samples or an inline size-exclusion chromatography setup.


1. Introduction

Small angle scattering (SAS) provides information on the shape of macro-molecules on the nanometer scale and is particularly suited for biological samples thanks to a large range of suitable buffer conditions. Unlike single crystal diffraction or NMR which offer atomic resolution, bioSAXS provides only information on the envelope of macro-molecules: it allows to validate the relative position of large structures and the assembly of biological complexes (Ubbink *et al.*, 2018). Structural biologists perform SAXS experiments to validate the size and the shape of their protein or complex under study. Since most beam-line users are neither synchrotron, nor SAXS specialists, a fully automated analysis of their data is of crucial importance for them. Automated data analysis should provide them not only with the reduced and pre-analyzed data, but also with all metadata and parameters needed to get their results published according to the relevant guidelines (Trehella *et al.*, 2017). These metadata are also important to reprocess data using third party software to ensure comparable results.

The BioSAXS beam-line from the European synchrotron (Pernot *et al.*, 2013) had an automated pipeline for the data-analysis which was based on EDNA (Incardona *et al.*, 2009) and was using the ATSAS (Petoukhov *et al.*, 2012) software underneath. While the outcome of the processing was very appreciated by beam-line users, the system was already close to the maximal throughput possible in terms of performances. The new EBS source (Chaize *et al.*, 2018) of the ESRF not only provides a higher brilliance, but also new wiggler sources for the former bending-magnet beam-lines which triggered the re-build and the upgrade of most of them. The BioSAXS beam-line (BM29) has been rebuilt in 2019 and features a 2-pole wiggler, a new flight-tube with an upgraded Pilatus 2M detector (planned to be) mounted in vacuum. This has led to a substantial increase of data-rate; mainly due to the new detector, twice larger than the former one.

This paper is divided in two main parts, starting with a presentation of the tools used for processing SAS data (*FreeSAS*) and for assembling pipelines (*dahu*). Then three different pipelines are presented: a common one used for the reduction of the scattering images and ~~two dedicated~~: one for sample-changer-based experiments (referred as SC) and ~~another one~~ dealing with chronological data used with the size exclusion chromatography experiments (referred as HPLC).

2. Tools

To meet the real-time data analysis requirements, all software used at the beam-line has been upgraded. Precise benchmark of the execution times of the previous EDNA-based pipeline demonstrated that most time was spent in launching external tools coming from the ATSAS suite and in parsing text files produced by those tools. It was decided to rewrite all pipeline in plain Python (van Rossum, 1989) and call SAS-related tasks from a library, *FreeSAS*. Finally  this code is interfaced to the control software, BLISS (Guijarro *et al.*, 2020), via Tango (Götz *et al.*, 2003) and uses a simple task scheduler, *dahu*, already used in production on the TruSAXS beam-line (Narayanan *et al.*, 2022).

2.1. Small angle scattering analysis tools, *FreeSAS*

FreeSAS is a Python library containing SAS-analysis tools available both via command line interface and from a Python API. It does not claim to be as complete as the ATSAS counterpart (Manalastas-Cantos *et al.*, 2021), but is free, released under the MIT license (i.e. it can be included in commercial products), all source code is available publicly on github (Kieffer *et al.*, 2021) and open to external contributions. Despite Python being an interpreted language, *FreeSAS* is performance oriented and most of the processing is performed with Cython (Behnel *et al.*, 2011) extensions writ-

4

ten in C to obtain the required performances. *FreeSAS* has been made available and packaged independently from *dahu* (the online analysis tools) so that scientists can reprocess their data and compare their results with those of other analysis software. The latest release, *FreeSAS* 0.9, supports Python 3.6 to 3.9.

2.1.1. SAS plotting: The *freesas* tool provides a way to plot a semi-logarithmic representation of the SAS curve: $I(q)$, where $q = 4\pi \sin(2\theta/2)/\lambda$ is the amplitude of the scattering vector and $I(q)$ the recorded intensity at a given q , along side some basic analyses which are described in the next sections.

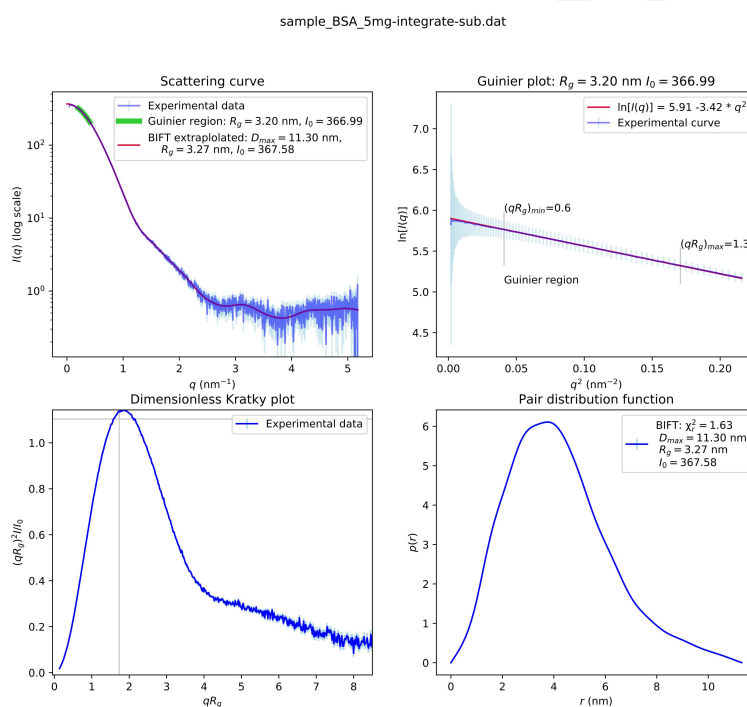



Fig. 1. Default visualization of bioSAS data with automated analysis

2.1.2. Guinier-region fitting: The first analysis performed on bioSAXS data determines the radius of gyration (R_g) of the solvated macro-molecule and the forwards

scattering intensity, I_0 (Guinier, 1994). Based on the Taylor expansion of the scattering curve at $q = 0$, R_g is obtained from the linear regression of $\log(I(q)) = \log(I_0) - 1/3(R_g q)^2$ on the proper q -range at small angles, called the Guinier Range. The selection of the Guinier-region is far from being obvious due to the beam-stop shadow, aggregation effects, and its upper limit depends on R_g itself: $q \cdot R_g < 1.3$. Thus multiple implementation are provided: *autorg*, which derives from the implementation in BioXTAS-Raw (Nielsen *et al.*, 2009); *auto-guinier*, which searches for a consensus region rather than the best possible Guinier-region; and finally *auto-gpa*, which performs a Guinier-peak-analysis (Putnam, 2016). The later is a quick assessment of the R_g and I_0 , sometimes less robust than the two other implementations, but suitable when many datasets are to be analyzed like in chromatography mode. It is worth mentioning that none of the three algorithms provide the exact same results as the original AUTORG (Petoukhov *et al.*, 2012) version from ATSAS. This highlights the importance of publishing the actual implementation of the algorithms with the associated numerical parameters used in it.

2.1.3. Pair distribution function: Although the scattering curve $I(q)$ is the Fourier transform of the pair distribution function $p(r)$, the later cannot directly be obtained from an inverse Fourier transform (IFT) due to the loss of phase information and the limited amount of information in the scattering curve. This ill-posed mathematical problem has no exact solution and is usually inverted with some extra constraints imposed, like the finite size of the support (defined by the maximum diameter, D_{max}). FreeSAS proposes an IFT based on the Bayesian statistics and derived from BIFT (Vestergaard & Hansen, 2006), ~~like the one found in~~ Nielsen *et al.* (2009). Despite  different approach, the result of *bift* is similar to the DATGNOM (Petoukhov *et al.*, 2007) from ATSAS which uses a Tikhonov's regularization. The diameter found, D_{max} , is

directly comparable with the one provided by ATSAS, but the parameter α differs since the theory used for regularisation differs.

2.1.4. Assessment of equivalence of scattering curves: To obtain the best possible signal from the sample, the capillary is exposed multiple times with a short exposure time. All equivalent frames are then merged to optimized the signal/noise ratio without pollution from radiation damaged data. The equivalence of a couple of frames is obtained from the correlation-map (CorMap) algorithm implemented from the publication by Franke *et al.* (2015).

2.1.5. Overlay of bead models: *subpycomp* (Brennich *et al.*, 2016), a tool to rotate/flip bead models and overlay them prior to merging them. It is equivalent to the SUPCOMB (Kozin & Svergun, 2001) tool from ATSAS.

2.1.6. Extraction to ASCII data: with the increased frame-rate of the new beam-line, the historical 3-column text file with q , I_{avg} , $\sigma(I)$ became unpractical and has been superseded by the hierarchical-data format, HDF5 (The HDF Group, 2000-2021) which contains all analysed data derived from raw data. The *extract-ascii* tool is designed to generate text files in the q , I_{avg} , $\sigma(I)$ format and offer a compatibility with third-party tools.

2.2. The job-manager: dahu

The role of the job manager is to ensure all processing requested by the client (i.e. the user interface) are actually performed, informs the client about the finished jobs and warns it in case of an error.

EDNA was used as workflow manager for the previous data-analysis pipeline for

many protein crystallography beam-lines (Incardona *et al.*, 2009) and for the BioSAXS beam-line at the ESRF (Brennich *et al.*, 2016). The parallelization model used in EDNA is based on Python threads and forking processes which was wasting resources in serializing and inter-process communication.

The *dahu* job-manager was designed for the TruSAXS beamline (Narayanan *et al.*, 2022) with those limitation in mind. The tango interface (Götz *et al.*, 2003) was kept similar to the one in EDNA to ease the adoption. The scheduling of jobs is performed via a shared queue and only few workers are running simultaneously in threads. Thus the code runs actually in parallel only in sections where the Global Interpreter Lock from Python (GIL) is released, like in Cython extensions (Behnel *et al.*, 2011) from FreeSAS or in the OpenCL code from pyFAI (Kieffer & Ashiotis, 2014).

2.2.1. Dahu-jobs manage the execution of the *plugins* (see here-after), provide an unique identifier which gives access to the status and output of the processing. Jobs see their input and output saved to the disk, this allows off-line reprocessing in case of issue during online data-analysis.

2.2.2. Dahu-plugins implement the processing logic of the different pipelines. Written in simple Python and fairly independent from the *dahu* framework, those plugins are often written or modified by beam-line scientists themselves.

2.2.3. Offline re-processing is made possible by the *dahu-reprocess* command-line tool. This tool was designed to (re-)execute one or several jobs based on the description saved by the online data analysis server. Since *dahu* has virtually no dependencies, it can be deployed on any computer to reprocess data. Nevertheless, to reprocess data acquired at the BioSAXS beam-line, one would need FreeSAS and all the other

8

dependencies of BioSAXS *plugins*, which are all documented, publicly available and all open-source.

3. Data analysis pipelines

There are two main experiments performed at the BioSAXS beam-line, either using the sample changer or the inline-chromatography setup (HPLC). Thus, two analysis pipelines were built, one for each of those experimental modes. The common part, mainly dealing with azimuthal integration, ~~got~~ integrated into a pre-processing pipeline called *integrate multi-frame*.

Since the Pilatus ~~3M~~ detector is controlled by the LIMA software (Petitdemange *et al.*, 2018), raw images are saved in a HDF5 file-format (The HDF Group, 2000-2021) with ten or hundred frames per file (depending on the acquisition mode, figure 2). HDF5 offers compression, faster data-access, symbolic links to datasets from one file to another ... but it is dramatically different from the former pipeline which was triggered frame per frame.

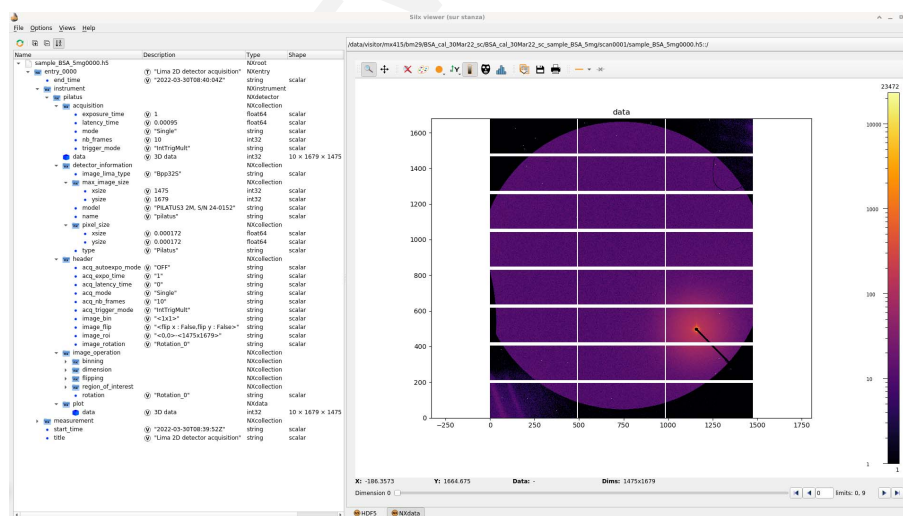


Fig. 2. Layout of a raw image HDF5-file produced by the LIMA acquisition software, visualized with the *silx* viewer.

ESRF provides several tools to visualize those HDF5 files like *silk view* (Vincent *et al.*, 2021) which is a graphical user interface based on Qt (Summerfield, 2007) or the web-viewer *h5web* (Bocciarelli *et al.*, 2022) to visualize those files inside a web-browser. This *h5web* is already used in the beam-line control user interface BsxCuBE3 (Tully *et al.*, 2022) and will be used in the new ISPyB interface (under development). LIMA files (figure 2) contain no metadata beside the camera configuration. Thus all sample and experiment description (geometry, mask, . . .), beam-stop diode intensities and other processing parameters have to be provided by the experiment sequencer, BLISS, as part of the job description when triggering the process.

The versatility of the HDF5 format allows to have one single output file for all results produced by a processing pipeline, making backup easier. Each pipeline registers the result of every individual processing step of the pipeline in the output HDF5 file (as HDF5-groups), together with the configuration associated with the processing. Input datasets are referenced using external links, which avoids data duplication while keeping traceability. Finally, metadata describing the sample, its buffer and the configuration of the beam-line are also recorded using the Nexus convention (Könnecke *et al.*, 2015). Each processing pipeline defines a default plot which tries to summarize the experimental result to the user.

3.1. Multi-frame integration pipeline

The multi-frame integration pipeline is triggered with the name of one LIMA-file (containing several frames) and all additional metadata describing the sample and the experiment. All other processing parameters are sent in the same way, regardless if they are read from the user-interface or collected by BLISS.

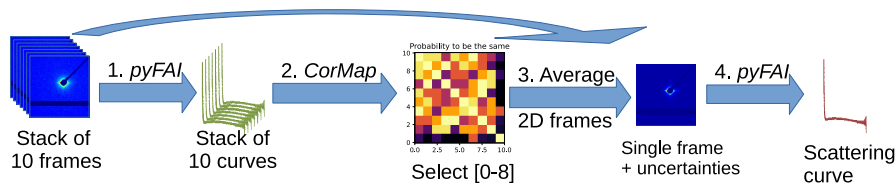


Fig. 3. Schematic of the multi-frame integration pipeline: 1. azimuthal integration of individual frames; 2. comparison of 1d-curves; 3. equivalent frames are averaged; 4. azimuthal integration of the averaged image.

The figure 4 presents a file produced by this processing pipeline with the default plot consisting in the semi-logarithmic representation of the scattering curve $I(q)$ viewed with *silx*. This pipeline is built of four subsequent analysis steps, as illustrated in figure 3:

1. Each recorded image is azimuthally integrated with *pyFAI* (Kieffer *et al.*, 2020) to produce one scattering curves per frame;
2. Scattering curves are compared, searching for radiation damage using the *CorMap* algorithm (Franke *et al.*, 2015); the probability for each pair of curves to be the same is compared to thresholds to assess their equivalence, those thresholds depend if frames were adjacent or not;
3. Equivalent images are averaged pixel-wise, weighted by the beam-stop diode intensity; variance is assessed using Poisson law;
4. The averaged frame is finally azimuthally integrated and uncertainties propagated accordingly.

11

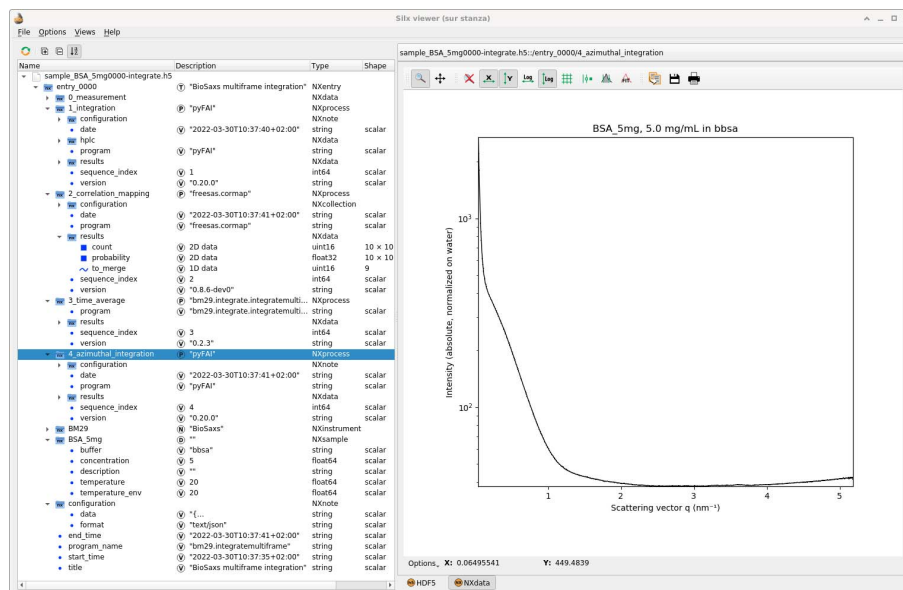


Fig. 4. Layout of a HDF5-file obtained from the multi-frame integration pipeline, visualized with the *silx* viewer. On the left-hand side, one can recognize in the HDF5-tree the structure of the pipeline described figure 3. The right-hand side presents the scattering curve of BSA before subtraction of background signal.

The plot of the azimuthally integrated averaged frame (at stage 4) is set as the default display when processing data in sample-changer mode. In HPLC-mode, the default plot represents the summed intensity as function of time, which is a fraction of the complete chromatogram. The HDF5-file additionally includes external links to the raw frames as acquired by the detector (stage 0).

It is worth mentioning that time-averaging and azimuthal-integration are not (strictly) commutative as demonstrated in appendix B, especially when it comes to uncertainty propagation.

3.2. Sample-changer pipeline

In sample-changer mode, solution containing samples are acquired alternatively with pure buffer solutions. The throughput of the beam-line is then limited by the pipetting system of the robot and the delay for cleaning the exposure chamber. The processing

is triggered with integrated data from the sample (i.e. the name of the file containing the sample data after azimuthal integration) and a list of buffer files corresponding to the different acquisition of buffers, usually the buffer before and the buffer after the sample acquisition.

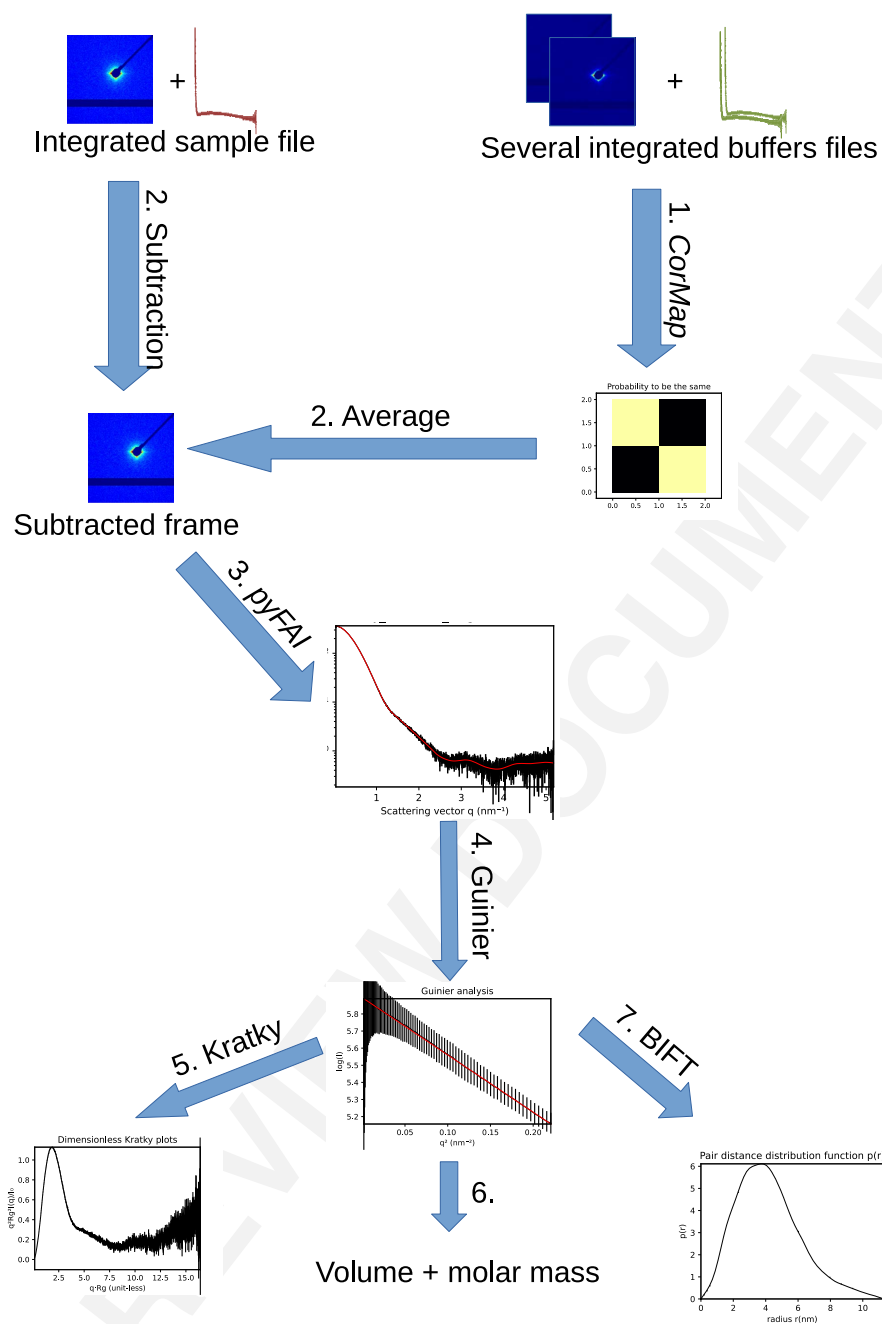


Fig. 5. Schematic of the sample-changer pipeline: 1. Comparison of buffer frames image; 2. Subtraction of averaged buffer image from sample image; 3. Azimuthal integration the background subtrated image; 4. and subsequent: SAS analysis, including Guinier, Kratky and BIFT analysis.

The sample-changer pipeline is schematized in figure 5 and produces a new HDF5-

file with the subtracted data in it, such file is visualized in figure 6 and contains the results of this 8-stage pipeline:

1. Comparison of buffer curves using the *CorMap* algorithm (Franke *et al.*, 2015);
2. Buffer frames are averaged together and subtracted from the sample averaged frame (i.e. still in 2D);
3. Azimuthal integration of the subtracted frame with *pyFAI* (Kieffer *et al.*, 2020);
4. Guinier analysis with the associated linear regression of $\log(I(q))$ vs. $\log(I_0) - 1/3(R_G q)^2$ at low q as default plot;
5. Dimensionless Kratky plot: $(q \cdot R_g)^2 \cdot I/I_0$ vs. $q \cdot R_g$;
6. Porod (Glatter & Kratky, 1982) and Rambo-Tainer invariant (Rambo & Tainer, 2013) calculation to assess the the molecular volume and molecular mass of the sample;
7. Indirect inverse Fourier transform using the BIFT algorithm (Vestergaard & Hansen, 2006) provides the pair distribution function $p(r)$;
8. Transfer of reduced data to ISPyB for BioSAXS (compatibility layer with legacy ISPyB).

As in the *multi-frame* processing pipeline (3.1), there is a link to the source data as stage zero of the processing to ensure a perfect tracking of the experiment. The pair distribution function obtained from BIFT (stage 7) allows to calculation of the radius of gyration in real space and should confirm the radius of gyration found from Guinier fit (stage 4).



The final stage of this pipeline is to register those results into the ISPyB for BioSAXS (De Maria Antolinos *et al.*, 2015) database and instantly ~~made~~ available to the user via the BioSAXS data portal (<https://exi2.esrf.fr>). The same data are shared with the BsxCuBE3 control software via a memcached key-value database for instant feed-back.

Online purification of the sample allows to reduce the effect of oligomerization. It has become a standard procedure since it was introduced to BM29 in 2012 (Round *et al.*, 2013) and accounts now for two third of all measurements performed at the beam-line.

IUCr macros version 2.1.17: 2022/03/29

frames. The input for this pipeline a list of HDF5-files with partial chromatograms integrated by the *multi-frame* pipeline presented in section 3.1. The HPLC pipeline, figure 7, re-builds the complete chromatogram and performs the complete analysis of the different fractions, taking into account the possibility for empty sections (due to missing input files). This pipeline produces files which are presented in figure 8.

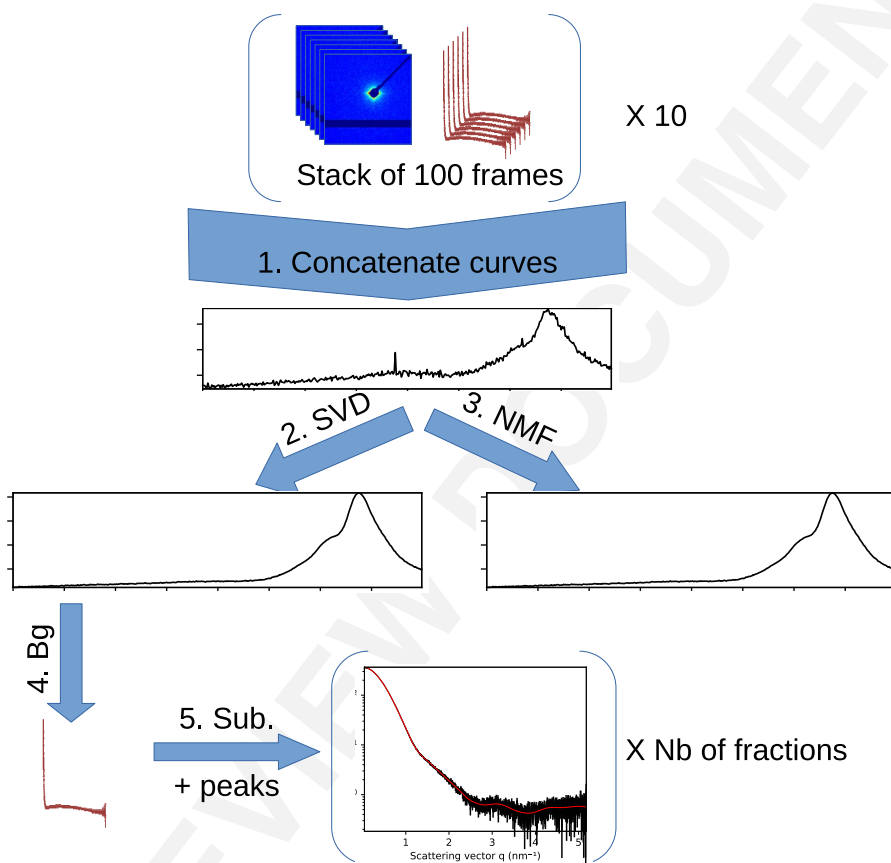


Fig. 7. Schematic of the HPLC pipeline: 1. Concatenate ; 2.&3. Multivariate analysis; 4. Background extraction; 5. Peak finding 6. SAS analysis on each fraction

This chromatography pipeline has six stages:

1. Concatenate partial chromatograms (1D curves) provided by the *multi-frame* pipeline to obtain the full chromatogram; Empty/missing regions are handled here;

17

2. Perform a Singular Value Decomposition (SVD) on the chromatogram to assess the number of components and extract scattering from the the background;
3. Perform a non-negative matrix factorisation (NMF) to ~~get an idea of the~~ scattering curve of the different fractions;
4. Select points belonging to the background by comparing experimental scattering with the first singular-vector from the SVD; average-out selected curves;
5. Perform peak-picking on subtracted curves; analyse each fraction with a similar pipeline to the one presented in 3.2
6. Prepare the data and send them to ISPyB for BioSAXS.

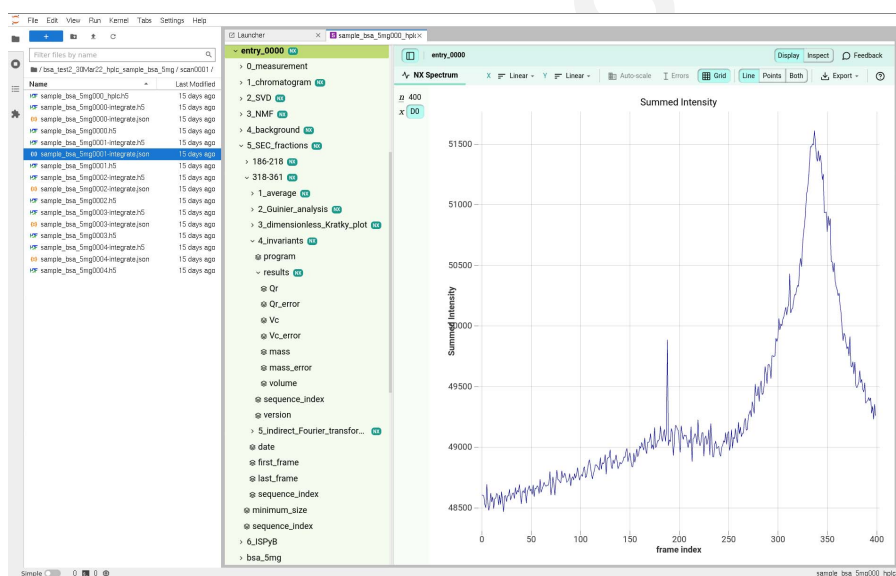


Fig. 8. Default visualization of the HDF5-file produced by the HPLC pipeline with the *h5web* viewer integrated into JupyterLab.

Multivariate analysis is performed to extract the signal of the macro-molecule from background scattering, and provide a hint on how many components have been separated in the chromatography. Gavish & Donoho (2014) provides the number of singular values/vectors which are to be saved after the SVD decomposition The SVD is

provided by the NumPy package (Oliphant, 2007). The first singular vector contains mainly background scattering, the following ones contains signal from the separated components and subsequent ones account only for noise. Since SVD (provided by NumPy (Oliphant, 2007)), does not enforce positivity of extracted singular vectors, a NMF decomposition is performed (provided by scikit-learn (Varoquaux *et al.*, 2015)).

Since none of the multivariate analysis propagates uncertainties, all processing needs to be re-done: a correlation-map is built between the first singular vector of the SVD and all experimental scattering curves. Those curves are ranked from the most likely to be pure buffer to the least likely. Since the major part of the collected fraction are expected to be buffer, the thirty percent of the curves which are the most similar to the first singular-vector are considered to be buffer and averaged together in stage 4. Uncertainties are propagated based on deviations calculated during azimuthal integration (and not on 2D frames, see appendix B).

The fraction collection (stage 5) is performed on the total scattering chromatogram, smoothed by median filtering. A peak search is performed with the *find_peaks_cwt* function from the *scipy* library (Jones *et al.*, 2001). It provides a list of region of high intensity scattering: the different fractions of the chromatogram. All subtracted curves from the same fraction are averaged and analysed with a similar pipeline as described in section 3.2: Guinier analysis, Kratky-plot, various invariant extraction and pair-distribution via BIFT. The results are presented the same way as in the sample-changer mode, one per fraction. The criteria for the fraction selection being intentionally soft, it is common that empty fractions are selected and that some or even all analysis fail on them.

4. Discussions

4.1. Statistics

The processing described in section 3 ~~has been put in~~ production in September 2020 and is operating for 20 months at the time of writing. The table 1 summarizes some figures collected:

Processing pipeline	#calls	Frame processed	run-time per job
Integrate multi-frame	42575	1230k	2.1s (1s, 10s)
Subtract & SAS analysis	11214	336k	7.3s
HPLC analysis	709	893k	2.9s

Table 1. Statistics of the number of job run over 20 months

The run-time for *multi-frame* integration presents a clear bi-modal distribution since the same code is used in sample-changer mode (10 frames/acquisition) and in HPLC mode where 100 frames are acquired per file. From those figures, one can estimate HPLC-mode represents 6% of the measurement performed but accounts for 72% of the total measurement time.

4.2. Performances

All computations are executed on a single computer equipped with a single hexa-core processor and an entry-level graphics card (Intel Xeon E5-2643 v3 + Nvidia Quadro M2000). During those 20 months, the *dahu*-server has been started 90 times, which corresponds to the weekly restart to apply security patches and other bug-corrections. Since all data-reduction occurs within the same process using threads, this demonstrates not only the reliability of the *dahu*-server but also of the whole pipeline including *FreeSAS* analysis.

4.3. Outlook

The foreseeable future should replace of the legacy version of the ISPyB database with a new ~~one~~ which will include better web visualization capabilities of the generated HDF5 files. The buffer averaging and subtraction in HPLC mode is not (strictly) exact

since it is based on integrated curves which have been normalized. It should be possible to weight properly those curves to obtain an average which is exactly the same as if one would have averaged or subtracted 2D frames and integrated the result (discussed in appendix B). Future algorithmic work will focus on *ab-initio* shape reconstruction, based on the DENSS (Grant, 2018) which is currently too slow to run with real-time constraints at the beam-line.

5. Conclusion

This document introduces the *FreeSAS* and the *dahu* software packages which are used respectively to analyse bioSAXS data and control online data analysis at the BioSAXS beam-line at the European synchrotron. Those two packages are combined with others to provide complete data-analysis pipelines. The three pipelines described in this contribution are used in production since 2020, and provide real-time feed-back of ongoing experiments to the user. All metadata, all parameters and all references to the source data are recorded together with the processed data into single HDF5 files which offers not only convenient storage but allows also reproducible science following the FAIR principle.

Appendix A Method

All figures were obtained from test-samples used at the beam-line: Bovine Serum Albumin (BSA) in solution at 5 mg/mL in a HEPES buffer.

Appendix B Rational for working with 2D frames rather than integrated curves

The *multi-frame* and *subtraction* pipelines are performing signal averaging and subtraction on 2D frames rather than azimuthally integrated curves. On the other hand, the HPLC pipeline performs the average and background subtraction on integrated curves.

This section describes the code for performing those two operations and discusses the rationale behind it. It will also try to distinguish which is the most correct. Let *frames* and *diode* be a list of acquired frames and beam-stop diode intensities, respectively. And assume *ai* is a configured azimuthal integrator (as available from pyFAI).

B1. Integrate before averaging

The code can be represent in those four lines of Python code:

```
integrated = [ ai.integrate1d(frame, npt, normalization_factor=norm, error_model="poisson")
               for frame, norm in zip(frames, diode)]
q = integrated[0].radial
I = numpy.mean([i.intensity for i in integrated], axis=0)
sigma = numpy.sqrt(numpy.sum([i.sigma**2 for i in integrated], axis=0))/len(integrated)
```

Intensities are obtained from an arithmetic average of already weighted curves. Uncertainties are obtained from a quadratic average of uncertainties propagated during integration. This method does not take into account the fact that some curves did have more weight than others during integration. The numerical values are correct only if normalisation factors were similar within the ensemble of curves.

B2. Average before integrating

The code can be represent in those three lines of Python code:

```
img_sum = numpy.sum(frames, axis=0)
nrm_sum = sum(diode)
q, I, sigma = ai.integrate1d(img_sum, npt, normalization_factor=nrm_sum, error_model="poisson")
```

The *img_sum* and *nrm_sum* are equivalent to a single exposure of the detector with much longer integration time, thus the larger normalization factor. Like this,

the contribution of the different frames are properly weighted during the azimuthal integration.

The non equivalence of the two approaches can be demonstrated when using completely arbitrary normalization factors (for example ranging from 1 to 10) for a set of equivalent images. The discrepancy is especially visible in the propagated uncertainties.

B3. Limitations and future improvements

In sample-changer mode, each pipeline handles only few dozens of images at a time, thus all data can easily be held in the memory of the computer. In HPLC mode, where several hundreds of frames are processed for a single measurement, the same technique could see the computer run out of memory. This is why the HPLC pipeline still averages integrated curves even if it not (strictly) correct.

In pyFAI, azimuthal integration is performed with ratios of sum of signals and sum of normalizations (Kieffer *et al.*, 2020). Those sums can be seen as partially processed data and those partial sums can be aggregated to obtain properly weighted average with their associated uncertainties, as described in Schubert & Gertz (2018). Thus, the memory consumption issue could be alleviated by saving not only the averaged signal, but also the sum of signals, the sum of normalizations and the partial variances.

Acknowledgements: The authors wish to thank Guillaume Bonamis for his former contribution to the FreeSAS library and Jesse Hopkins from APS for the fruitful discussion and the the sharing of the code from BioXTAS-RAW.

References

- Behnel, S., Bradshaw, R., Citro, C., Dalcin, L., Seljebotn, D. & Smith, K. (2011). *Comput. Sci. Eng.* **13**(2), 31–39.
- Bocciarelli, A., Huder, L. & Vincent, T., (2022). silx-kit/h5web: v4.0.0.
<https://doi.org/10.5281/zenodo.6458453>
- Brennich, M. E., Kieffer, J., Bonamis, G., De Maria Antolinos, A., Hutin, S., Pernot, P. & Round, A. (2016). *Journal of Applied Crystallography*, **49**(1), 203–212.
<https://doi.org/10.1107/S1600576715024462>

- Chaize, J. *et al.* (2018). In *Proc. of International Conference on Accelerator and Large Experimental Control Systems (ICALEPCS'17)*, Barcelona, Spain, 8-13 October 2017, no. 16 in International Conference on Accelerator and Large Experimental Control Systems, pp. 2010–2015. Geneva, Switzerland: JACoW. <https://doi.org/10.18429/JACoW-ICALEPCS2017-FRAPL07>.
<http://jacow.org/icalepcs2017/papers/frapl07.pdf>
- De Maria Antolinos, A., Pernot, P., Brennich, M. E., Kieffer, J., Bowler, M. W., Delageniere, S., Ohlsson, S., Malbet Monaco, S., Ashton, A., Franke, D., Svergun, D., McSweeney, S., Gordon, E. & Round, A. (2015). *Acta Crystallographica Section D*, **71**(1), 76–85.
<http://dx.doi.org/10.1107/S1399004714019609>
- Franke, D., Jeffries, C. M. & Svergun, D. I. (2015). *Nat Meth*, **12**(5), 419–422.
- Gavish, M. & Donoho, D. L. (2014). *IEEE Transactions on Information Theory*, **60**(8), 5040–5053.
- Glatter, O. & Kratky, O. (1982). *Small Angle X-ray Scattering*. Blackwell Science.
<https://books.google.fr/books?id=6TNSxgEACAAJ>
- Götz, A., Taurel, E., Pons, J., Verdier, P., Chaize, J., Meyer, J., Poncet, F., Heunen, G. & Götz, E. (2003). In *Proceedings of ICALEPCS2003*. Gyeongju, Korea. MP705.
- Grant, T. D. (2018). *Nature Methods*, **15**(3), 191–193.
<https://doi.org/10.1038/nmeth.4581>
- Guijarro, M., Berruyer, G., Beteva, A., Claustre, L., Coutinho, T., Dominguez, M., Guillo, P., Guillo, C., Homs, A., Meyer, J., Michel, V., Pancino, P., Papillon, E., Perez, M., Petitdemange, S., Pithan, L., Sever, F. & Valls, V. (2020). In *Proc. ICALEPCS'19*, no. 17 in International Conference on Accelerator and Large Experimental Physics Control Systems, pp. 70–77. JACoW Publishing, Geneva, Switzerland. <https://doi.org/10.18429/JACoW-ICALEPCS2019-MOCPL03>.
<https://jacow.org/icalepcs2019/papers/mocpl03.pdf>
- Guinier, A. (1994). *X-ray diffraction in crystals, imperfect crystals, and amorphous bodies*. New York: Dover.
- Incardona, M.-F., Bourenkov, G., Levik, K., Pieritz, R., Popov, A. & Svensson, O. (2009). *J. Synchrotron Rad.* **16**(6), 872–879.
- Jones, E., Oliphant, T. E. & Peterson, P., (2001). SciPy: Open source scientific tools for Python.
<http://www.scipy.org/>
- Kieffer, J. & Ashiotis, G. (2014). In *PROC. OF THE 7th EUR. CONF. ON PYTHON IN SCIENCE (EUROSCIPY 2014)*.
<http://arxiv.org/pdf/1412.6367.pdf>
- Kieffer, J., Bonamis, G. & Brennich, M. E., (2021). FreeSAS: Open source small angle scattering tools for Python. Zenodo.
<https://doi.org/10.5281/zenodo.4463031>
- Kieffer, J., Valls, V., Blanc, N. & Hennig, C. (2020). *Journal of Synchrotron Radiation*, **27**(2), 558–566.
<https://doi.org/10.1107/S1600577520000776>
- Könnecke, M., Akeroyd, F. A., Bernstein, H. J., Brewster, A. S., Campbell, S. I., Clausen, B., Cottrell, S., Hoffmann, J. U., Jemian, P. R., Männicke, D., Osborn, R., Peterson, P. F., Richter, T., Suzuki, J., Watts, B., Wintersberger, E. & Wuttke, J. (2015). *Journal of Applied Crystallography*, **48**(1), 301–305.
<https://doi.org/10.1107/S1600576714027575>
- Kozin, M. B. & Svergun, D. I. (2001). *Journal of Applied Crystallography*, **34**(1), 33–41.
<http://dx.doi.org/10.1107/S0021889800014126>
- Manalastas-Cantos, K., Konarev, P. V., Hajizadeh, N. R., Kikhney, A. G., Petoukhov, M. V., Molodenskiy, D. S., Panjkovich, A., Mertens, H. D. T., Gruzinov, A., Borges, C., Jeffries, C. M., Svergun, D. I. & Franke, D. (2021). *Journal of Applied Crystallography*, **54**(1), 343–355.
<https://doi.org/10.1107/S1600576720013412>

- Narayanan, T., Sztucki, M., Zinn, T., Kieffer, J., Homs-Puron, A., Gorini, J., Van Vaerenbergh, P. & Boesecke, P. (2022). *Journal of Applied Crystallography*, **55**(1), 98–111.
<https://doi.org/10.1107/S1600576721012693>
- Nielsen, S. S., Toft, K. N., Snakenborg, D., Jeppesen, M. G., Jacobsen, J. K., Vestergaard, B., Kutter, J. P. & Arleth, L. (2009). *Journal of Applied Crystallography*, **42**(5), 959–964.
<http://dx.doi.org/10.1107/S0021889809023863>
- Oliphant, T. E. (2007). *Comput. Sci. Eng.* **9**(3), 10–20.
- Pernot, P., Round, A., Barrett, R., De Maria Antolinos, A., Gobbo, A., Gordon, E., Huet, J., Kieffer, J., Lentini, M., Mattenet, M., Morawe, C., Mueller-Dieckmann, C., Ohlsson, S., Schmid, W., Surr, J., Theveneau, P., Zerrad, L. & McSweeney, S. (2013). *Journal of Synchrotron Radiation*, **20**(4), 660–664.
<http://dx.doi.org/10.1107/S0909049513010431>
- Petitdemange, S. *et al.* (2018). In *Proc. of International Conference on Accelerator and Large Experimental Control Systems (ICALPCS'17), Barcelona, Spain, 8-13 October 2017*, no. 16 in International Conference on Accelerator and Large Experimental Control Systems, pp. 886–890. Geneva, Switzerland: JACoW. <https://doi.org/10.18429/JACoW-ICALPCS2017-TUPHA194>.
<http://jacow.org/icalpcs2017/papers/tupha194.pdf>
- Petoukhov, M. V., Franke, D., Shkumatov, A. V., Tria, G., Kikhney, A. G., Gajda, M., Gorba, C., Mertens, H. D. T., Konarev, P. V. & Svergun, D. I. (2012). *Journal of Applied Crystallography*, **45**(2), 342–350.
<http://dx.doi.org/10.1107/S0021889812007662>
- Petoukhov, M. V., Konarev, P. V., Kikhney, A. G. & Svergun, D. I. (2007). *Journal of Applied Crystallography*.
- Putnam, C. D. (2016). *Journal of Applied Crystallography*, **49**(5), 1412–1419.
<https://doi.org/10.1107/S1600576716010906>
- Rambo, R. P. & Tainer, J. A. (2013). *Nature*, **496**(7446), 477–481.
<http://dx.doi.org/10.1038/nature12070>
- van Rossum, G., (1989). Python programming language.
<http://www.python.org>
- Round, A., Brown, E., Marcellin, R., Kapp, U., Westfall, C. S., Jez, J. M. & Zubieta, C. (2013). *Acta Crystallographica Section D*, **69**(10), 2072–2080.
<http://dx.doi.org/10.1107/S0907444913019276>
- Schubert, E. & Gertz, M. (2018). In *Proceedings of the 30th International Conference on Scientific and Statistical Database Management, SSDBM '18*. New York, NY, USA: Association for Computing Machinery.
<https://doi.org/10.1145/3221269.3223036>
- Summerfield, M. (2007). *Rapid GUI Programming with Python and Qt*. Prentice Hall, 1st ed.
- The HDF Group, (2000–2021). Hierarchical data format version 5.
<http://www.hdfgroup.org/HDF5>
- Trehwella, J., Duff, A. P., Durand, D., Gabel, F., Guss, J. M., Hendrickson, W. A., Hura, G. L., Jacques, D. A., Kirby, N. M., Kwan, A. H., Pérez, J., Pollack, L., Ryan, T. M., Sali, A., Schneidman-Duhovny, D., Schwede, T., Svergun, D. I., Sugiyama, M., Tainer, J. A., Vachette, P., Westbrook, J. & Whitten, A. E. (2017). *Acta Crystallographica Section D*, **73**(9), 710–728.
<https://doi.org/10.1107/S2059798317011597>
- Tully, M. D., Kieffer, J., Oscarsson, M., Florial, J. B., Beteva, A., Popov, A., Moussaoui, D., Brennich, M. E., Aberdam, R. C., Papp, G., McCarthy, A., Mueller-Dieckmann, C., Leonard, G. & Pernot, P. (2022). *Journal of Synchrotron Radiation*.
- Ubbink, M., Perrakis, A., Schroer, M. A. & Svergun, D. I. (2018). *Emerging Topics in Life Sciences*, **2**(1), 69–79.
<https://doi.org/10.1042/ETLS20170138>
- Varoquaux, G., Buitinck, L., Louppe, G., Grisel, O., Pedregosa, F. & Mueller, A. (2015). *GetMobile: Mobile Comp. and Comm.* **19**(1), 29–33.
<https://doi.org/10.1145/2786984.2786995>

Vestergaard, B. & Hansen, S. (2006). *Journal of Applied Crystallography*, **39**(6), 797–804.
<https://doi.org/10.1107/S0021889806035291>

Vincent, T., Valls, V., Payno, H., Kieffer, J., Solé, V. A., Paleo, P., Naudet, D., de Nolf, W., Knobel, P., Garriga, J., Retegan, M., Rovezzi, M., Fangohr, H., Kenter, P., UUSim, Favre-Nicolin, V., Nemoz, C., Picca, F.-E., Caswell, T. A., Bertoldo, J. P. C., Campbell, A., Wright, C. J. C., Communie, G., Kotanski, J., Coutinho, T., N0B0dY, Kim, S.-W., schooft, Farago, T. & linupi, (2021). silx-kit/silx: 1.0.0: 2021/12/06. Zenodo.
<https://doi.org/10.5281/zenodo.5761269>

Synopsis

Detailed presentation of the automatic data analysis pipelines for the BioSAXS beam-line at the European synchrotron.
