EUCALL

The European Cluster of Advanced Laser Light Sources

Grant Agreement number: 654220

Work Package 4 – SIMEX

Deliverable D4.4
Simulated coherent scattering data from plasma and non–plasma samples

Lead Beneficiary: HZDR, XFEL

Carsten Fortmann-Grote, Jan-Philipp Burchert, Michael Bussmann, Marco Garten, Axel Hübl, Thomas Kluge, and Adrian P. Mancuso

Due date: September 30, 2017
Delivery date: September 29, 2017

Project webpage: www.eucall.eu

Deliverable Type	
R = Report DEM = Demonstrator, pilot, prototype, plan designs DEC = Websites, patents filing, press & media actions, videos, etc. OTHER = Software, technical diagram, etc.	R
Dissemination level	
PU = Public, fully open, e.g. web CO = Confidential, restricted under conditions set out in Model Grant Agreement CI = Classified, information as referred to in Commission Decision 2001/844/EC	PU

This project has received funding from the *European Unions Horizon 2020 research and innovation programme* under grant agreement No 654220.

1. Outline

As first applications of our simulation software suite *simex_platform* [1, 2, 17], we have employed the simulation capabilities of *simex_platform* [1] to simulate two photon experiments:

1. Coherent diffraction from high–power laser excited optically thin and thick plasmas (plasma sample)
2. Single particle imaging at the European XFEL (non–plasma sample)

In addition, we also report on the integration of detector simulations in *simex_platform*.

2. Plasma sample

In Deliverable Report D4.3 [17], we present the interoperability of X–ray Free–Electron Laser (XFEL) and Ultra–High Intensity (UHI) laser pulse generation, interaction with the target, and generation of a Small–Angle X–ray Scattering (SAXS) image using the scattering code *ParaTAXIS*.

A Particle In Cell (PIC) simulation provides the time evolution of the electron density on which the XFEL photons are scattered. We assume invariance of the target in propagation direction and simulate the XFEL pulse with 10^{12} photons for which the target is optically thin, see left part of Fig. 1. We further demonstrate scattering on an optically thick setup by simulating resonant scattering on the ions of the target. The ion density follows the electron density as expected for plasma expansion into vacuum[3], see right part of Fig. 1.

The signal of the optically thick target is washed out due to higher scattering probability. In both, the optically thin and the optically thick case, the SAXS pattern well resolves the nanometer–scale grating depth and period, taking into account the target evolution during the interaction time with the laser pulse.

All density data from the PIC simulation as well as the SAXS patterns are published on Zenodo together with the data format documentation [20] in partial fulfillment of EUCALL Milestone M4.3 [18].

3. Molecular (non–plasma) sample

Application of *simex_platform* to single–particle imaging experiments at the European X–ray Free Electron Laser (SPB–SFX scientific instrument) has been published as Ref. [4]. It is also the basis for an online tutorial on the wiki pages of *simex_platform* and it is discussed in the EUCALL Milestone M4.2 [19] (First example simulation). The datasets for coherent diffraction from the protein 2NIP are deposited on the EUCALL Data Repository [21].

The signal generation for non–plasma samples happens in two steps: First, the photon–matter interaction module calculates for each time step of the simulation the atomic positions and the atomic form factors for each atom in the sample in the field of the x–ray laser. The effect of radiation damage can be switched on or off in the simulation. In the latter case, the atoms remain on their initial positions and the form factors correspond to the atomic ground states. This information is then taken in the second step to calculate the scattered intensity as a function of time for each pixel in the virtual 2D pixel detector. The software package *pysingfel* [5] implements the

This project has received funding from the *European Unions Horizon 2020 research and innovation programme* under grant agreement No 654220.

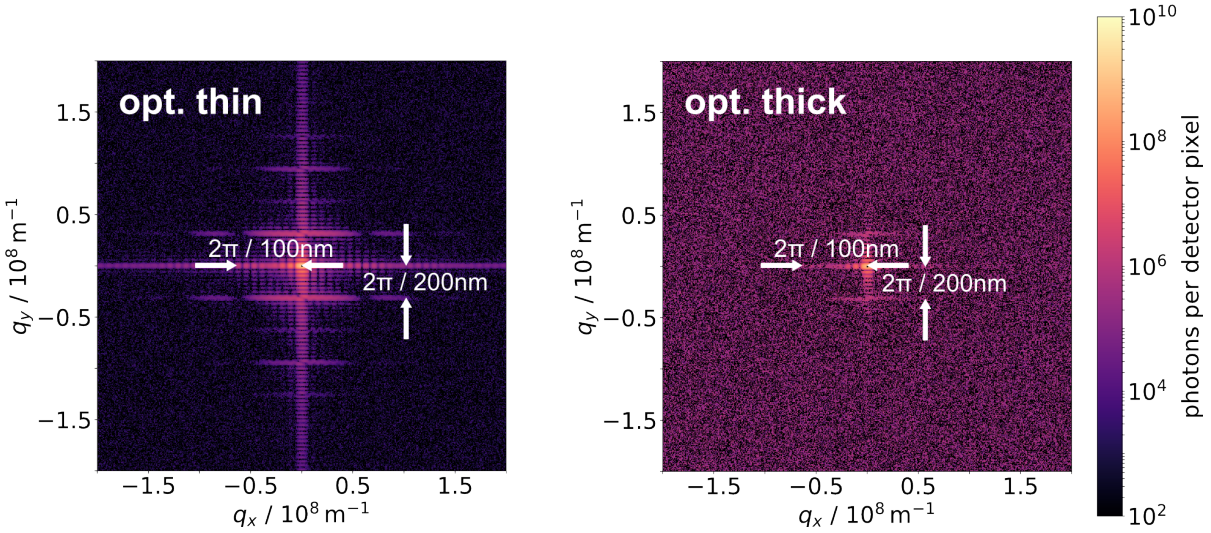


Figure 1: **Left:** *ParaTAXIS* SAXS image for the optically thin target at 1.4 m distance from the target, detector pixel size a_D = 13.5 µm, X-ray wavelength λ_{XFEL} = 1.47 Å and 10^{12} photons in the illuminated area. The vertical separation of scattering lines corresponds to the grating period of 200 nm, the horizontal to the grating depth of 100 nm. **Right:** *ParaTAXIS* SAXS image for the optically thick target. Here, the scattering cross section was increased by a factor of 1000 to account for resonant scattering at the ion density. All other parameters remain the same.

scattering formula

$$I(\mathbf{q}) = \Omega \frac{d\sigma_{\mathrm{Th}}(\theta)}{d\Omega} \langle I_0 \rangle \left| \sum_i f_i(\mathbf{q}) e^{i\mathbf{q}\cdot\mathbf{R_i}} \right|^2 , \tag{1}$$

where $d\sigma_{\mathrm{Th}}/d\Omega$ is the differential Thomson cross section, $\langle I_0 \rangle$ is the average pulse intensity, Ω is the solid angle spanned by the considered detector pixel. The wavevector $\mathbf{q}$ depends on the detector geometry (distance d from the sample) and pixel coordinates (r_x, r_y) in the detector plane assumed to be perpendicular to the beam propagation axis and centered such that the beam axis intersects the detector plane at its origin:

$$q = \frac{2\pi}{\lambda} \begin{pmatrix} \sin(2\theta)\cos(\phi) \\ \sin(2\theta)\cos(\phi) \\ \cos(2\theta) - 1 \end{pmatrix} = \frac{2\pi}{\lambda} \begin{pmatrix} \frac{r_x}{\sqrt{r_x^2+r_y^2+d^2}} \\ \frac{r_y}{\sqrt{r_x^2+r_y^2+d^2}} \\ \frac{d-\sqrt{r_x^2+r_y^2+d^2}}{\sqrt{r_x^2+r_y^2+d^2}} \end{pmatrix} . \tag{2}$$

Here, $2\theta = \arctan\left(\frac{r_x^2+r_y^2}{d}\right)$ is the scattering angle and $\phi = \arctan\left(\frac{r_y}{r_x}\right)$. In this notation, the pixel solid angle becomes $\Omega = 4\arcsin\left(a^2/\left[4(r_x^2 + r_y^2 + d^2) + a^2\right]\right)$, with the pixel width a.

In our simulations, d = 13 cm. The detector is a 512 × 512 array of pixels of side length a = 200 µm corresponding to one quadrant of a 1 megapixel AGIPD detector [6] at the SPB–SFX instrument at European XFEL.

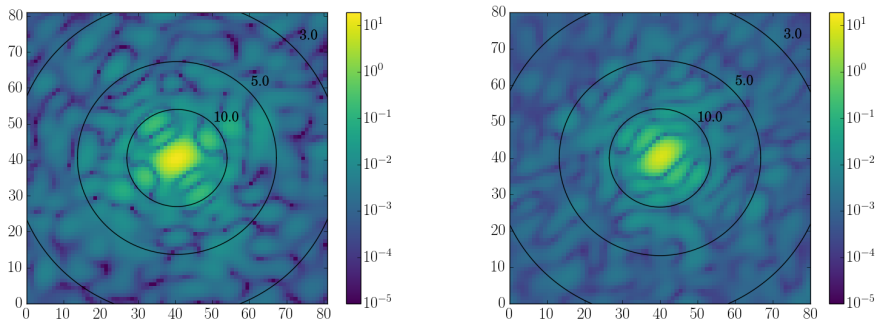
We have applied this simulation toolchain to two cases:

1. Diffraction of 5 keV photons from isolated 2NIP molecules and variation of the XFEL pulse duration, published in Ref. [4].

This project has received funding from the *European Unions Horizon 2020 research and innovation programme* under grant agreement No 654220.

2. Diffraction of 5 keV photons from hydrated 2NIP molecules and variation of the hydration layer thickness, published in Ref. [7].

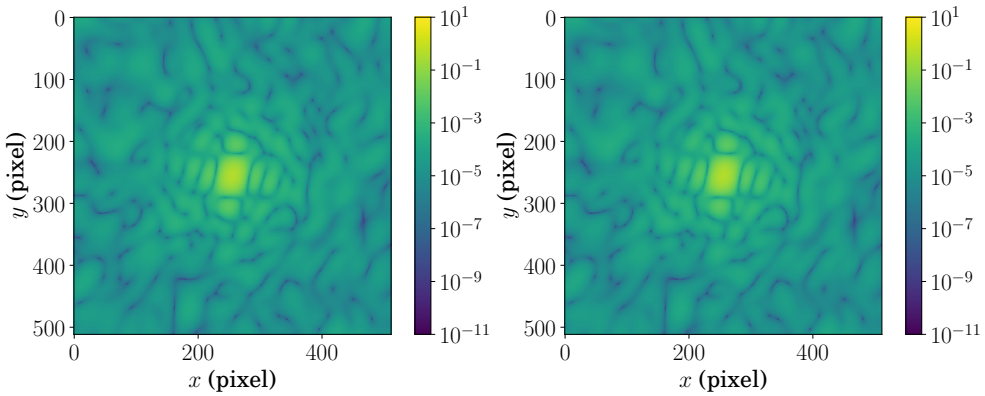
Fig. 2 shows diffraction patterns from 2NIP with and without radiation damage taken into account in the simulation. Fig. 3 shows diffraction from the same molecule for two different hydration layer



(a) no radiation damage (b) with radiation damage

Figure 2: Diffraction from 2NIP with 5 keV photons without (a) and with (b) radiation damage taken into account. Taken from Ref. [7].

thicknesses of 3 Å (a) and 10 Å (b), respectively.



(a) 3 Å hydration layer (b) 10 Å hydration layer

Figure 3: Diffraction patterns for 5 keV XFEL photons scattering from a single 2NIP molecule embedded in a hydration shell of 3 Å (a) and 10 Å (b) thickness.

It can be seen how, qualitatively, the presence of imperfections (radiation damage or hydration layer) influences the quality of diffraction signals, e.g. the speckle contrast and noise level. In both cases, we performed statistical analysis of the simulated diffraction patterns as a function of the varied parameters pulse duration and hydration layer thickness, respectively to gain more quantitative insight into the dependency of diffraction data quality on these effects.

Fig. 4 shows the coefficient of variation, a measure of interpretability of single–particle diffraction patterns [8, 4] for Free–Electron Laser (FEL) pulse durations of 3 fs, 9 fs and 30 fs. The larger

This project has received funding from the *European Unions Horizon 2020 research and innovation programme* under grant agreement No 654220.

coefficients of variation found in the 3 fs case indicating poor data quality is due to the significantly lower photon flux at these ultra–short pulses. This in turn comes from the intrinsic dependence of pulse duration on the electron bunch charge FEL: Highly charged bunches, producing a high photon flux, experience higher space charge effects and stronger longitudinal dispersion leading to longer pulse durations.

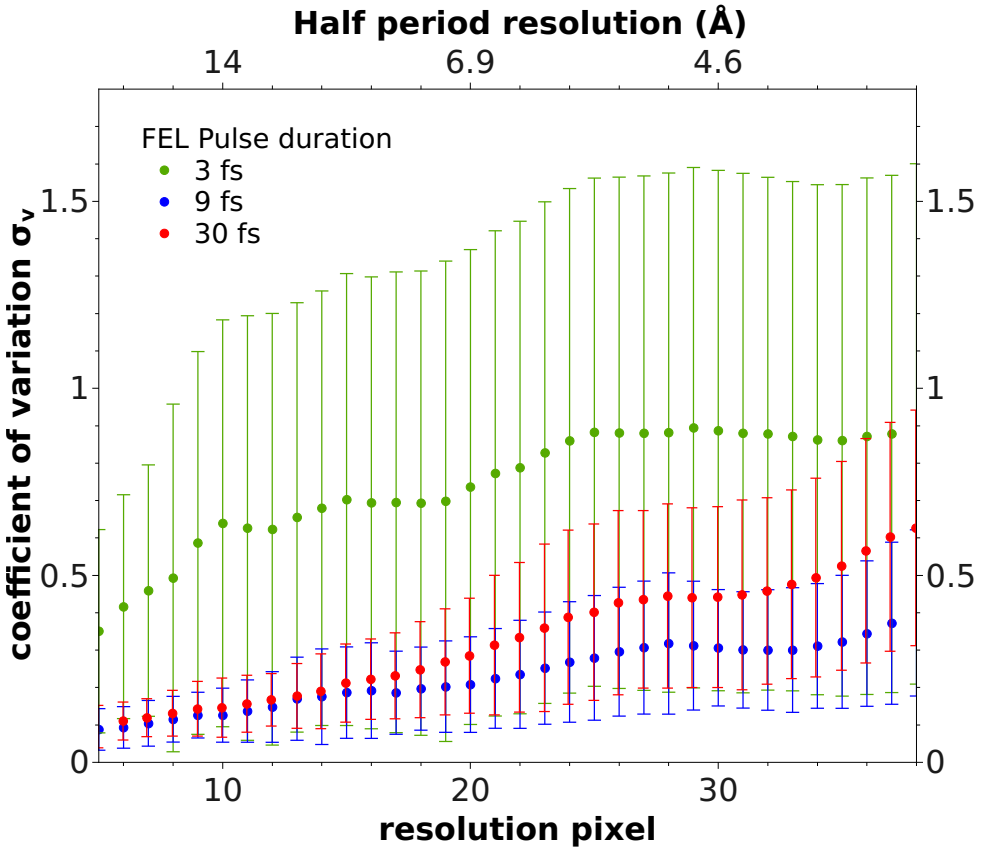


Figure 4: Coefficient of variation obtained from oriented simulated diffraction patterns for 2NIP probed by 5 keV XFEL pulses of varying pulse duration and photon flux. Figure taken from Ref. [4].

Fig. 5 shows the R–value for averaged simulated diffraction patterns as a function of the hydration layer thickness. Radiation damage has been neglected in these simulations to isolate the effect of the water presence. $R < 0.2$ indicates good data quality allowing reconstruction of the 3D electron density at resolution $d = 2\pi/q$. Following this argument, the presence of a 3 Å thick hydration layer would allow reconstruction on the level of a few Å length scales.

4. Detector simulations

Simulations of measurable signals should not only take into account the mechanism of x–ray scattering from the sample but also the response of the detector to the scattered light impinging on the detector surface. So far, these detector simulations have been missing in *simex_platform* for various technical reasons as discussed e.g. in the EUCALL mid term report. These obstacles have

This project has received funding from the *European Unions Horizon 2020 research and innovation programme* under grant agreement No 654220.

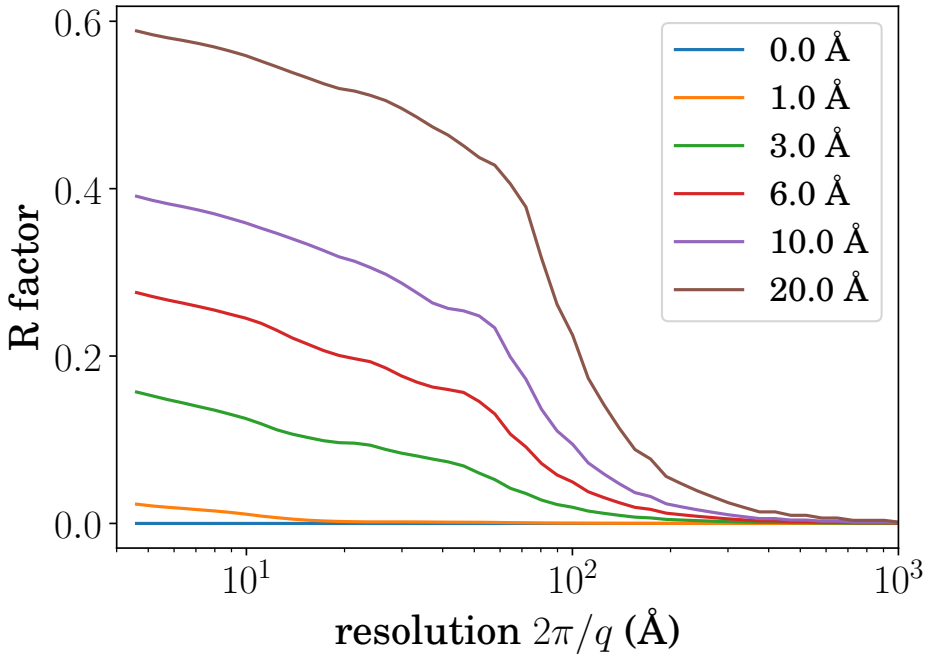


Figure 5: *R*–value obtained from simulated diffraction patterns for 2NIP probed by probed by 5 keV XFEL pulses of 9 fs pulse duration as a function of the hydration layer thickness. Figure taken from Ref. [7]

now been overcome. In the following, we present the implementation of detector simulations in the simulation environment *simex_platform* followed by a first example simulation.

4.1. Detector simulation software

For the purpose of detector simulations, the software packages `Geant4` (`http://geant4.cern.ch/`) and *X-CSIT* (`https://git.xfel.eu/gitlab/karaboDevices/xcsit`) are utilized that have already been successfully implemented in the data analysis and control framework of the European XFEL *karabo* (`https://git.xfel.eu/gitlab/Karabo/Framework`). In contrast to *karabo*, where all the simulation is performed with devices which are integrated into the framework processing pipe, *SimEx* is a collection of classes. Consequently, installation as well as maintenance and usage of new components are more convenient in *SimEx*.

Another difference is the programming language: In contrast to *karabo* and *X-CSIT* which are written in C++, *SimEx* is written in python. Since *X-CSIT* is the basis of the simulation also in this project, the interface defined needs to be made accessible from python. To integrate it into *SimEx* an additional calculator needs to be written that utilizes the extended functions from *X-CSIT*. To achieve this, an interface between C++ written *X-CSIT* and python written *SimEx* source code is designed and implemented. This includes writing source code in C++ and python as well as creating a build procedure with *cmake*. Furthermore, an appropriate documentation and similar coding style like the one used for other *SimEx* calculators is required.

This project has received funding from the *European Unions Horizon 2020 research and innovation programme* under grant agreement No 654220.

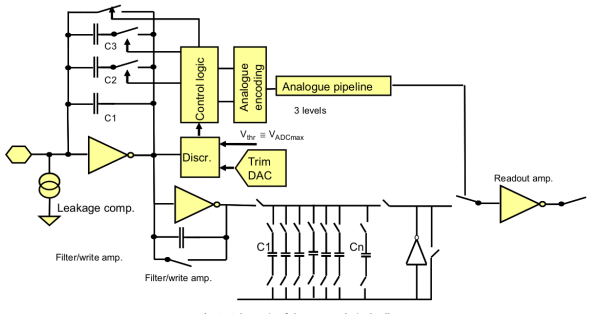


Figure 6: Application Specific Integrated Circuit of a single pixel of the AGIPD detector with storage pipeline. Source: [9].

4.2. Detector Effects

Particle detectors are complex electronic devices ranging from transistors to APplication specific Integrated Circuits (APICs) [10] [11]. Due to imperfections in the material those circuits bear a natural source of fluctuations. Especially, the influence of leakage currents on operation amplifiers that increase with radiation intensity can result in increased noise levels [11]. The resulting noise is often still smaller than e.g. the statistical fluctuations in the photon number in each pixel [11].

Furthermore, leakage currents from another origin are much more serious. Since the XFEL source produces more images than can be processed in one of its 600 µs pulse trains, the detector needs to store those image values previous to processing in a pipeline for each pixel (see figure 6). After each of those pulse trains there is a gap of approx. 100 ms, where the detector reads the pixel values of all stored images, digitizes them and store them on a hard disk. This pipelines consist of capacities and switches that suffer from leakage currents, as well. This not only affects the output values of intensity but also the noise level of the readout and digits [11].

Detectors at European XFEL have to cope with a very complex parameter space. They have to cover a broad range of energy of many keV for every pixel independently of each other. Furthermore, the pulse rate of the recording is quite high. Additionally, they need to be sensible enough to detect single photons but robust enough not to be destroyed at high intensities. This requires protection of the electronics leading to additional readout noise [10]. Despite optimizations, detectors have a finite dynamic range of values, which, once exceeded, leads cut–off signals [10].

Additionally, there are also physical effects producing noise. Thermal fluctuations can be reduced by cooling. However, high intensity radiation can create plasmas in detector pixels that effect neighbouring pixels due to electron drift [10]. The process is called “blooming” and is taken into account in the charge simulation of this detector simulation project. Additional information can be found in references: Joy et al. [12], Rüter et al. [13], Shi et al. [11] and Potdevin et al. [10].

Last but not least, the number of photons arriving at a pixel and the number of charges created from an interacting photon are statistical values. They are uncertain and follow the Poisson statistics. Together with the blooming this makes up the most important source for noise. Nevertheless, in many experiments the noise resulting from a well optimized detector is much less than the fluctuations in the signal resulting e.g. from optical elements[10].

This project has received funding from the *European Unions Horizon 2020 research and innovation programme* under grant agreement No 654220.

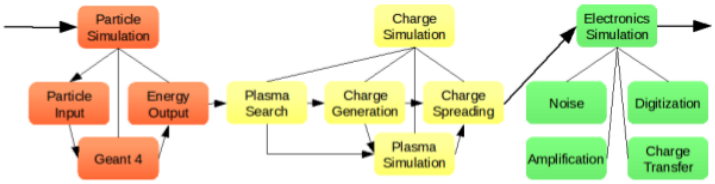


Figure 7: Schema of the different parts of the detector simulation software *X–CSIT*. Source: [12].

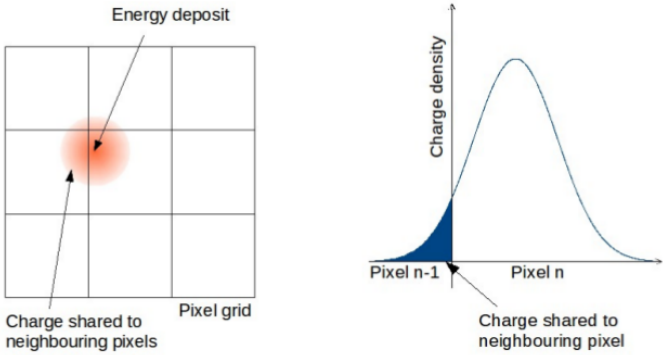


Figure 8: Sketch how a charge cloud produced from a single interaction can spread to neighbouring pixels. Source: [12].

4.3. Existing Software

4.3.1. Geant4/ X–CSIT

For the simulation of detectors Joy et al. [12] have created an object oriented software library in the C++ programming language, called *X–CSIT*[1], to simulate the behaviour of 2D semiconductor pixel detectors. Due to the object orientation not only the initally implmented LPD, AGIPD, DCCS, pnCCD and FastCCD detectors but also derived detectors can be supported [12]. Initially written for being integrated into the *karabo* framework, the software is universal enough to be stand–alone.

X–CSIT consists of three parts [12] as can be seen in figure 7. The first one, the particle simulation, describes how photons interact with the active layer of the detector. For this purpose *X–CSIT* acts as a wrapper of the interaction simulation software *Geant4* [12]. *Geant4* covers the physical models of a broad range of energy ranging from keV to TeV [14]. The models can handle electromagnetic processes such as the photo electric effect and fluorescence but also hadronic and optical processes. The standard processes such as photo electric effect, Compton and Rayleigh scattering, as well as Auger processes are also included [12, 14]. For this part *X–CSIT* has the task to manage the data transfer from and to *Geant4*. The second part, the charge simulation, deals with the propagation of the charges and plasmas created by interactions. Their behaviour is mainly governed by drift and diffusion of electrons which can be described with a Gaussian normal distribution (see figure 8) with the following standard derivation [12]:

$$\sigma_d = \sqrt{\frac{2k_B T}{qE} \cdot d}. \tag{3}$$

[1] *X–CSIT* is not free software, hence access is limited to users who have access to the European XFEL gitlab repository.

This project has received funding from the *European Unions Horizon 2020 research and innovation programme* under grant agreement No 654220.

Here *d* is the depth in the material, *T* the temperature, *q* the drifting total charge and *E* the electrical field that pulls the charge to the readout electronics of the detector.

The last part of *X–CSIT* covers the detector readout electronics. The electronic simulation simulates the electronic components of the detector and their behaviour. For this purpose, *X–CSIT* offers modules which are combined to represent the circuits of the detector [12]. This simulation is not included the detector simulations for *simex_platform* because it requires a tight connection to, e.g., the calibration software and database for specific detectors.

4.4. Extending C++ to Python

Since *X–CSIT* is written in C++ and the calculators of the *SimEx*[2] are written in python, there is a need of extending C++ classes to python. To extend the X–CSIT classes to python, we used the *boost.python* library (`http://www.boost.org/doc/libs/1_65_0/libs/python/doc/html/index.html`) *Boost.python* consits of header files that need to be added to the C++ implementations. Additionally, in a BOOST_PYTHON_MODULE needs to be defined. This module creates the equivalent of a python module and needs to define all the extended classes, their functions as well as their return and input parameters explicitly [15]. Consequently, *boost.python* can be seen as a module including abstract types that can be linked to C++ instances as well as to python instances.

4.5. Design

Since C++ and python programming languages belong to the object oriented programming languages, the concept of inheritance and polymorphism are well suited to achieve the desired extension and integration. Polymorphism is the key concept for using self written classes of this design in *X–CSIT*. It allows also to minimize redundant source code that is always a potential source of error and very difficult to maintain. However, the required features remain accessible. For this reason, all the created classes except *Constants* are derived from *X–CSIT* or *SimEx* base classes and interfaces as can be seen in figure 9.

Another important aspect of this design is the design of the python class. In the end, this is the class that is accessed and used for performing the simulation. For this reason, it should have control over input and output as well as control over running the simulation. As can be seen in figure 9, the *XCSITPhotonDetector* calculator is given control over each step of the simulations. This includes creating data containers, initiate and run the simulations as well as reading the input file and creating the output file.

Nevertheless, the simulation itself should be triggered from C++ code. One reason for this is that tunnelling through the *boost.python* layer is assumed to be slow. Additionally, python code is slower than C++ and there where additional features needed such as a changed function signature in *ChargeSim*. Consequently, setting up and running simulations is programmed in C++ and only calling these functions is extended to python.

Last but not least, the entire project needs to be integrated into *simex_platform*. With regard to the source code style and function this can easily be achieved by applying inheritance. However, one does not want to compile *X–CSIT* each time you compile also this project. For this reason, for both *X–CSIT* and py_detector_interface *cmake* is used to compile and link the compiled classes to shared objects. Shared objects are the Unix equivalents of Windows' dynamical linked libraries (.dll) offering a comfortable way to release and use applications.

[2] *SimEx* is the python library inside the simulation environment *simex_platform*

This project has received funding from the *European Unions Horizon 2020 research and innovation programme* under grant agreement No 654220.

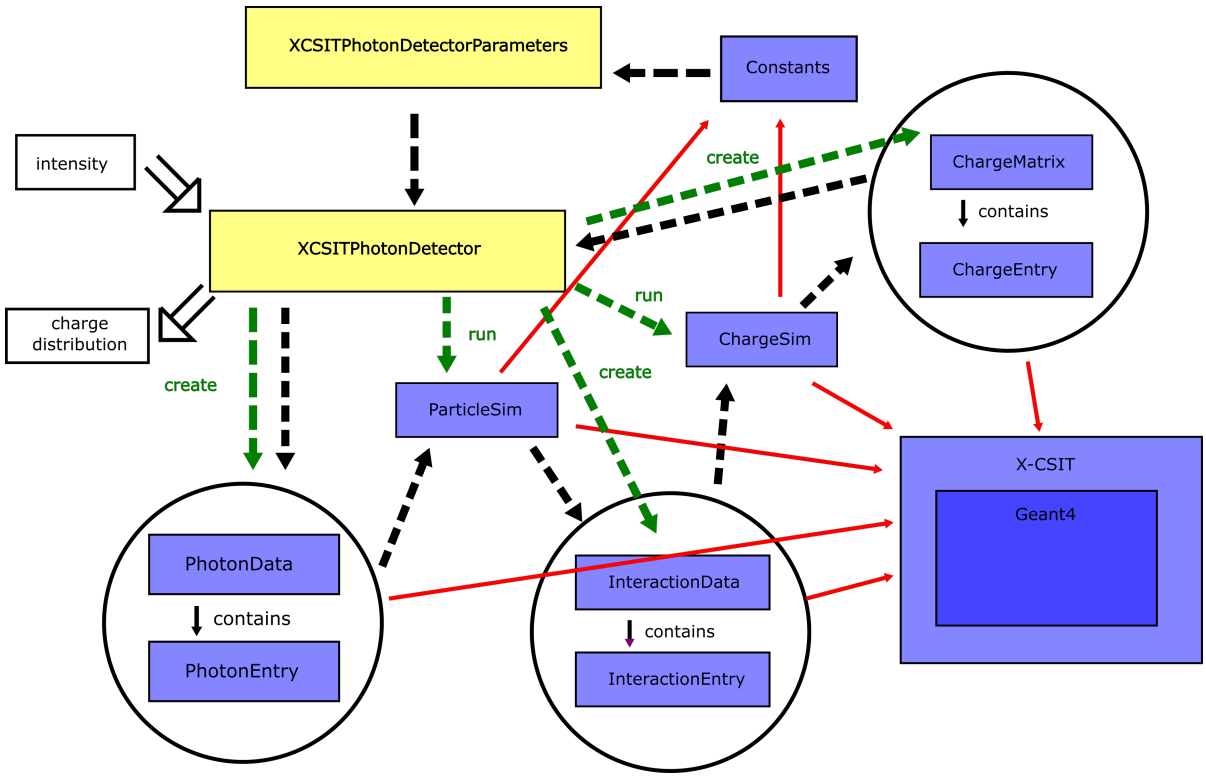


Figure 9: Sketch showing the dependencies and inheritance structure of the py_detector_interface. Colour code: yellow ≙ classes written in python, blue ≙ classes written in C++, red arrows ≙ inheritance, green arrows ≙ manipulation of other instances, black arrows ≙ data flow.

This project has received funding from the *European Unions Horizon 2020 research and innovation programme* under grant agreement No 654220.

category	options
DetectorType	pnCCD, LPD, AGIPD, AGIPDSPB, CAD
PlasmaSearch	BLANK
PlasmaSim	BLANKPLASMA
PointSim	FULL, FANO, LUT, BINNING

Table 1: This table show the options to choose from when selecting a mode for the simulations.

4.5.1. C++ classes

There are essentially two groups of C++ classes written for this project. The first one consists of the input and output containers. They are derived from abstract interfaces located in *X–CSIT*, which have already an implementation in *X–CSIT*. The following data containers where implemented using the abstract *X–CSIT* interfaces: *XPhotonEntry*, *XPhotonData*, *XInteractionData*, *XInteractionEntry*, *XChargeEntry* and *XChargeData*. Due to polymorphism other *X–CSIT* functions can deal with classes derived from them.

The second group of C++ classes deal with the simulations. There is a simulation of the photons interacting with the matter of the detector and a simulation dealing with the propagation of created charges in the detector. Both simulations have a parent class in *X–CSIT*. Their task is to behave like a filter. To run a simulation with the *X–CSIT* parent classes certain functions with certain formal parameters in their signature have to be called in a specific sequence. In order to avoid the need to extending all those types from *X–CSIT* possible to use as these formal parameters the simulation classes are necessary.

The functions of *ParticleSim* and *ChargeSim* receive strings to choose which instances of *X–CSIT* need to be instantiated and bound to a formal parameter of a *X–CSIT* simulation call. Furthermore, this make addition of e.g. detectors easier because they need to be added to the C++ simulation classes and *Constants* only. There is no need to export them to python. Currently the following options are included: Furthermore, the simulation characterisation options specified in table 1 are needed in various classes. For instance, the „DetectorType“option is required for both *ParticleSim* and *ChargeSim*. Additionally, all their constants need to be accessible from the *XCSITPhotonDetectorParamters* class as well. For this reason, the constants are stored in an own class. Exploiting the capacities of C++ classes to inherit from many parent classes, *ParticleSim* and *ChargeSim* are not just inheriting from their *X–CSIT* parents but also from *Constants*. In principle, it would also be possible to make *XCSITPhotonDetectorParamters* inherit the constants from *Constants*. Since this is much more complicated due to the nature of the attributes of *Constants* (arrays of strings) than adding functions to *Constants* that return the values, the latter was implemented.

4.5.2. Python calculators

Two python classes were written. The first one, *XCSITPhotonDetectorParameters*, implements the abstract python class *AbstractCalculatorParameters*. Its purpose is to gather and check all the input parameters. If an parameter is set which is not specified in *Constants* an exception is raised. Nevertheless, instances of this class are essentially containers with property getter and setter functions. The properties are the same as the options in table 1.

This project has received funding from the *European Unions Horizon 2020 research and innovation programme* under grant agreement No 654220.

The second class is the calculator itself. It implements *AbstractPhotonDetector* which itself is derived from *AbstractBaseCalculator*. For this reason, the way the simulation is performed is already predetermined:

1. After instantiation python calls immediately the init function. This function possesses three formal parameters: a *XCSITPhotonDetectorParameters* instance, a variable to store the path for the input file and a variable to store the path for the output file. Since python variables do not have types, the init function has to check if the inserted actual parameters fit with the required instances. Furthermore, init deals with incomplete input.

2. The next method to call is *readH5*. Since the input to this calculator is different to the input of *ParticleSim* and *ChargeSim*, the data from the hdf5 input file has to be translated: The matrix of intensities is read and transformed into instances of *PhotonEnty* stored in an instance of *PhotonData*. The instances of *PhotonEntry* store for each photon the following attributes: energy, normalized vector of flight direction, current position. Those values where calculated from the input data by applying geometry.

3. For running the simulation the *backengine* method has to be called. It consists of two parts:
   a) The *PhotonData* instance is passed into *ParticleSim* which transfers the container into *XCSIT::XGeant4ParticleSim* where interactions of the photons with the detector material are simulated. The output container
   InteractionData is also passed to those classes. During the simulation it is filled with instances of type *InteractionEntry* that contain for each interaction the deposited energy in the material at a given site of the detector and the time when that happens after start.
   b) The instance of *InteractionData* is handed to the instance of *ChargeSim* which transfers it into *XCSIT::XPlasmaPointChargeSim* and the *Geant4*
   classes respectively. Since the readout electronics cannot be at the surface of a detector an electrical field is applied to pull the created charges in the material to the readout electronics. During this propagation charge clouds resulting from e.g. plasmas can broaden and effect neighbouring pixels. This is simulated in *ChargeSim*. The output is an instance of *ChargeMatrix*, where each element, *ChargeEntry*, represents a pixel of the detector and each element contains the number of charges recorded in that detector pixel. Please note, that the perspective to look at the matrix is parallel to the z-axis/ propagation direction of the light.

4. Last but not least, the data containers and their content are written to the hdf5 output file at the location specified by the output path.

The structure of the input and output file can be found in the wiki of this project (https://github.com/eucall-software/py_detector_interface/wiki).

4.6. Application

To test the termination and as a proof of principle the tutorial (https://github.com/eucall-software/simex_platfrom/wiki/SimEx-Tutorial) has been used to create a sample diffraction pattern of Nitrogenase Iron protein from the *Protein Data Base* (see PDB 2NIP [16]). The used detector quadrant of the „AGIPD“detector has 512 x 512 pixels of 200 µm x 200 µm size. All the photons are simulated with an energy of 4.96 keV and the detector is 13 cm away from the origin of diffraction. To obtain

This project has received funding from the *European Unions Horizon 2020 research and innovation programme* under grant agreement No 654220.

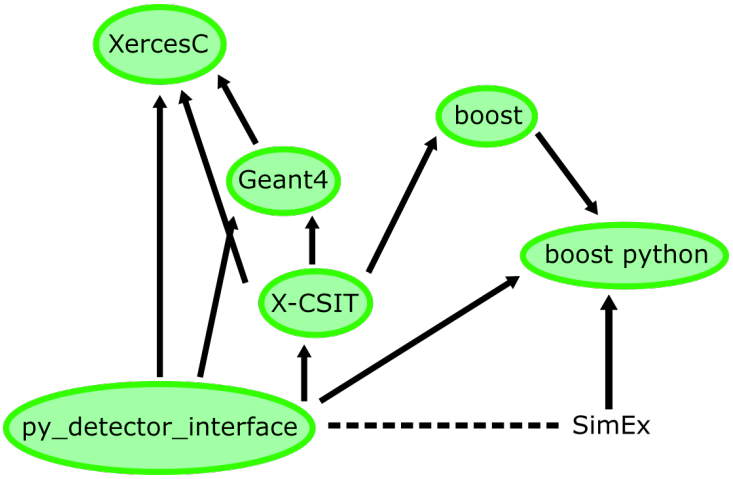


Figure 10: Depicted are the dependencies of the py_detector_interface project. The green circled names represent shared objects (.so) on Unix operating systems.

a visible detector response in reasonable compute times, the scattered intensity was multiplied by a factor 10^5. The preliminary results of this study are shown in in Fig. 11).

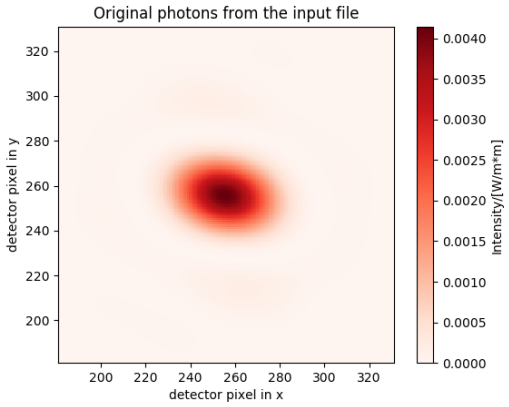
4.7. Conclusions

Detector simulations based on the existing library *X–CSIT* are now part of *simex_platform*. A proof–of–concept simulation demonstrates the workflow. The results will have to analyzed and the software needs to be profiled and debugged before the detector simulations can be used in production simulations. This will be among the upcoming tasks in the SIMEX work package.

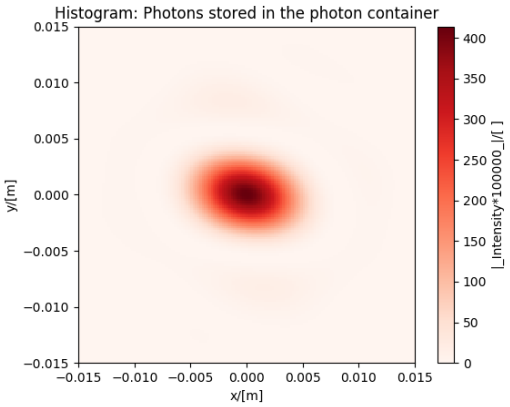
A. Deviation from proposal

As mentioned in the EUCALL mid term report, ELI's contributions to the SIMEX work package were re–defined: Within SIMEX, ELI focusses on the development of a simulation pipeline for Laser–Plasma Acceleration (LPA) based coherent light sources. The progress is presented in the Deliverable Report D4.3 [17]. The task "signal generation from plasma samples" of D4.4 was assigned to HZDR, and the task "signal generation from non–plasma samples" was assigned to XFEL.

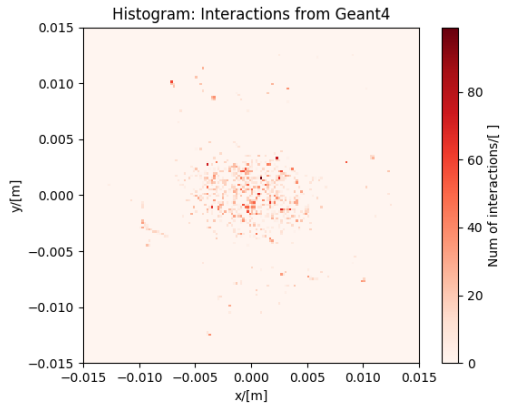
This project has received funding from the *European Unions Horizon 2020 research and innovation programme* under grant agreement No 654220.



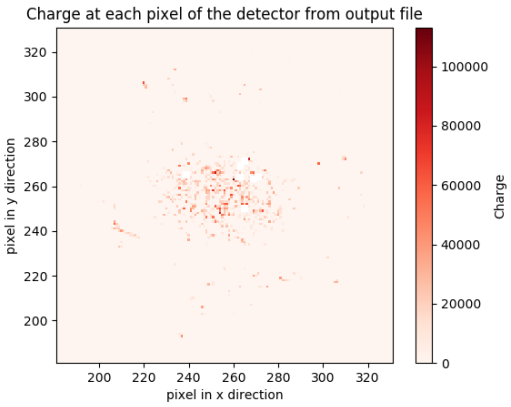
(a) Input intensity calculated by the diffraction calculator.



(b) Photonlist created from the the input (see 11a).



(c) Interactions of the photons with the detector material calculated by X–CSIT and Geant4.



(d) Output of the charge propagation simulation.

Figure 11: This figure shows the state of the data containers at intermediate steps of the simulation.

This project has received funding from the *European Unions Horizon 2020 research and innovation programme* under grant agreement No 654220.

Journal articles

1. *The github repository for simex_platform*, <https://github.com/eucall-software/simex_platform>.
2. Fortmann-Grote, C., Andreev, A. A., Appel, K., Branco, J., Briggs, R., Bussmann, M., Buzmakov, A., Garten, M., Grund, A., Huebl, A., Jurek, Z., Loh, N. D., Nakatsutsumi, M., Samoylova, L., Santra, R., Schneidmiller, E. A., Sharma, A., Steiniger, K., Yakubov, S., Yoon, C. H., Yurkov, M. V., Zastrau, U., Ziaja-Motyka, B., and Mancuso, A. P., *Simulations of ultrafast x–ray laser experiments*, in *Advances in X-ray Free-Electron Lasers Instrumentation IV*, SPIE Optics + Optoelectronics, **10237**, (International Society for Optics and Photonics, 2017), 102370S, doi:10.1117/12.2270552.
3. Mora, P., *Plasma Expansion into a Vacuum*, *Phys. Rev. Lett.* **90,** 185002 (2003).
4. Fortmann-Grote, C., Buzmakov, A., Jurek, Z., Loh, N.-T. D., Samoylova, L., Santra, R., Schneidmiller, E. A., Tschentscher, T., Yakubov, S., Yoon, C. H., Yurkov, M. V., Ziaja-Motyka, B., and Mancuso, A. P., *Start-to-end simulation of single-particle imaging using ultra-short pulses at the European X-ray Free-Electron Laser*, *IUCrJ* **4,** 560–568 (2017).
5. *The github repository for pysingfel (Single particle diffraction at FELs, python implementation)*, <https://www.github.com/eucall-software/pysingfel>.
6. Allahgholi, A., Becker, J., Bianco, L., Delfs, A., Dinapoli, R., Goettlicher, P., Graafsma, H., Greiffenberg, D., Hirsemann, H., Jack, S., Klanner, R., Klyuev, A., Krueger, H., Lange, S., Marras, A., Mezza, D., Mozzanica, A., Rah, S., Xia, Q., Schmitt, B., Schwandt, J., Sheviakov, I., Shi, X., Smoljanin, S., Trunk, U., Zhang, J., and Zimmer, M., *AGIPD, a high dynamic range fast detector for the European XFEL*, *J. Inst.* **10,** C01023–C01023 (2015).
7. Fortmann-Grote, C., Bielecki, J., Jurek, Z., Santra, R., Ziaja-Motyka, B., and Mancuso, A. P., *Simulations of single-particle imaging of hydrated proteins with x-ray free-electron lasers*, in *Advances in Computational Methods for X-Ray Optics IV*, SPIE Optical Engineering + Applications, **10388**, (International Society for Optics and Photonics, 2017), 103880M, doi:10.1117/12.2275274.
8. Yoon, C. H., Yurkov, M. V., Schneidmiller, E. A., Samoylova, L., Buzmakov, A., Jurek, Z., Ziaja, B., Santra, R., Loh, N. D., Tschentscher, T., and Mancuso, A. P., *A comprehensive simulation framework for imaging single particles and biomolecules at the European X-ray Free-Electron Laser.*, *Scientific reports* **6,** 24791 (2016).
9. Henrich, B., Becker, J., Dinapoli, R., Goettlicher, P., Graafsma, H., Hirsemann, H., Klanner, R., Krueger, H., Mazzocco, R., Mozzanica, A., *et al.*, *The adaptive gain integrating pixel detector AGIPD a detector for the European XFEL*, *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* **633,** S11–S14 (2011).
10. Potdevin, G., and Graafsma, H., *Analysis of the expected AGIPD detector performance parameters for the European X-ray free electron laser*, *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* **659,** 229–236 (2011).
11. Shi, X., Dinapoli, R., Henrich, B., Mozzanica, A., Schmitt, B., Mazzocco, R., Krüger, H., Trunk, U., Graafsma, H., Consortium, A., *et al.*, *Challenges in chip design for the AGIPD detector*, *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* **624,** 387–391 (2010).

This project has received funding from the *European Unions Horizon 2020 research and innovation programme* under grant agreement No 654220.

12. Joy, A., Wing, M., Hauf, S., Kuster, M., and Rüter, T., *X-CSIT: a toolkit for simulating 2D pixel detectors*, *Journal of Instrumentation* **10,** C04022–C04022 (2015).

13. Rüter, T., Hauf, S., Kuster, M., Joy, A., Ayers, R., Wing, M., Yoon, C. H., and Mancuso, A. P., *X-RAY DETECTOR SIMULATION PIPELINES FOR THE EUROPEAN XFEL*, in *IEEE Conference Proceedings of the Nuclear Science Symposium 2015*, (2016).

14. Agostinelli, S., Allison, J., Amako, K. a., Apostolakis, J., Araujo, H., Arce, P., Asai, M., Axen, D., Banerjee, S., Barrand, G., *et al.*, *GEANT4–a simulation toolkit*, *Nuclear instruments and methods in physics research section A: Accelerators, Spectrometers, Detectors and Associated Equipment* **506,** 250–303 (2003).

15. Abrahams, D., and Seefeld, S., *boost.python documentation*, <http://www.boost.org/doc/libs/1_65_0/libs/python/doc/html/index.html> (2017).

16. Berman, H., Westbrook, J., Feng, Z., Gilliland, G., Bhat, T., Weissig, H., Shindyalov, I., and Bourne, P., *The Protein Data Bank*, Nucleic Acids Research, 28: 235-242, <www.rcsb.org> (2000).

Project reports

17. Fortmann-Grote, C., Andreev, A., Sharma, A., Briggs, R., Garten, M., Huebl, A., Grund, A., Kluge, T., Yakubov, S., Bussmann, M., and Mancuso, A. P., *Deliverable D4.3, Interoperability of simulation workflows*, Project Deliverable report, (European Cluster of Advanced Laser Lightsources (EUCALL), 2017), doi:10.5281/zenodo.998647, <https://dx.doi.org/10.5281/zenodo.998647>.

18. Fortmann-Grote, C., *Milestone M4.3, Simulations interoperable*, Project Milestone report, (European Cluster of Advanced Laser Lightsources (EUCALL), 2017), doi:10.5281/zenodo.998654, <https://dx.doi.org/10.5281/zenodo.998654>.

19. Fortmann-Grote, C., *Milestone M4.2, First example simulation*, Project Milestone report, (European Cluster of Advanced Laser Lightsources (EUCALL), 2017), doi:10.5281/zenodo.998696, <https://dx.doi.org/10.5281/zenodo.998696>.

Datasets

20. Garten, M., Hübl, A., Grund, A., Fortmann-Grote, C., Kluge, T., and Bussmann, M., *ParaTAXIS X-ray Scattering Input & Output*, European Cluster of Advanced Laser Lightsources (EUCALL), <https://dx.doi.org/10.5281/zenodo.885033> (2017).

21. Fortmann-Grote, C., *Simulated coherent diffraction from 2NIP (SPB-SFX instrument, 3 fs, 4.96 keV European XFEL pulses)*, European Cluster of Advanced Laser Lightsources (EUCALL), <https://dx.doi.org/10.5281/zenodo.886087> (2017).

This project has received funding from the *European Unions Horizon 2020 research and innovation programme* under grant agreement No 654220.