EUCALL

The European Cluster of Advanced Laser Light Sources

Grant Agreement number: 654220

Work Package 4 – SIMEX

Deliverable D4.5
Testing, validation, and example workflow

Lead Beneficiary: DESY

Carsten Fortmann-Grote, Johannes Reppin, Frank Schlünzen, Yves Kempp, Sergey Yakubov, Ashutosh Sharma, Alexander Andreev, Sakura Pascarelli, Mathias Sander, Thomas Kluge, Marco Garten, Axel Hübl, Michael Bussmann, and Adrian P. Mancuso

Due date: September 30, 2018
Delivery date: September 25, 2018

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 Union's Horizon 2020 research and innovation programme* under grant agreement No 654220.

Abstract

This report details validation and benchmark studies of the experiment simulation capabilities developed in EUCALL's workpackage 4 (SIMEX). Where available, we compare our simulations to experimental data. In other cases, we compare simulations with simulations, using either different implementations of the same modeling approach or two simulations of varying degree of accuracy (e.g. 1D vs. 2D radiation hydrodynamics). We then summarize HPC benchmarks of selected simulation codes which present performance bottlenecks in the respective simulation pipeline. Finally, a simulation environment based on the jupyterHub technology is presented.

Contents

1. Validation

1.1. Single–particle coherent diffractive imaging

We compare a 3D molecular structure reconstructed from simulated diffraction data to the original 3D model reconstructed from NMR [1]. measurements.

Coherent diffraction from single proteins has been simulated using the code *singFEL*, which is integrated in *simex_platform*. Of the order 200000 diffraction images, sampling the FEL spectral and temporal fluctuations, including focussing mirror height profiles, radiation damage processes

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

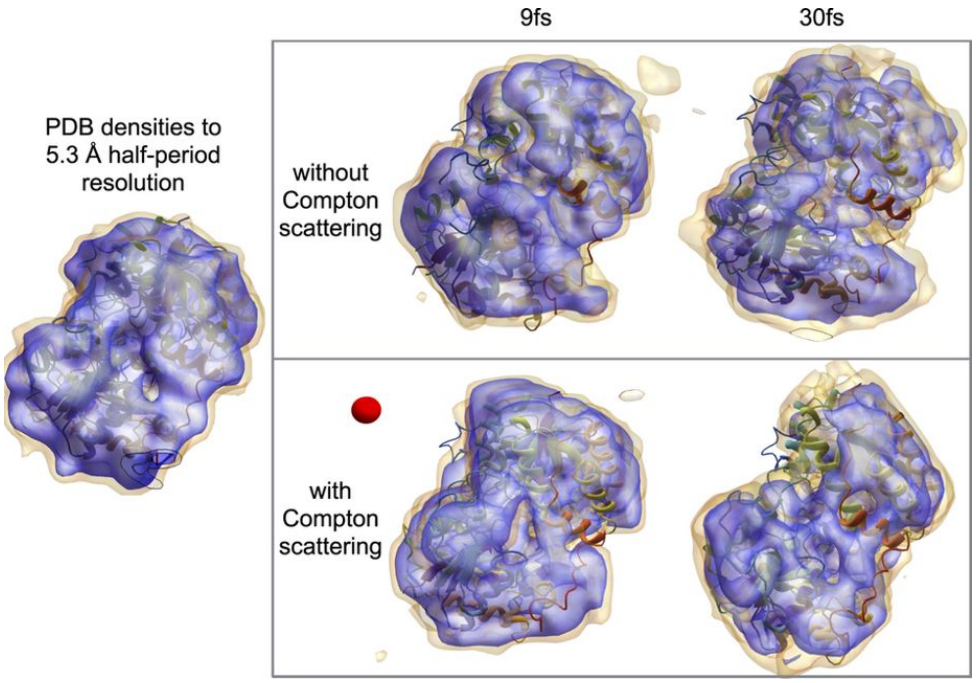


Figure 1: Electron densities of the average reconstruction at the 5% and 15% levels (yellow and purple, respectively) from simulated diffraction data with (top row) and without (bottom row) Compton scattering and for two different XFEL pulse durations. The protein's low–resolution features (original PDB shown on left-hand-side) were recovered in all four cases, with surface electron densities showing the most variation and hence least certainty. Loss of reproducible density is more severe in the 30 fs case due to greater damage. Degradation of surface contrast is expected in previous damage–only simulations and may, in future studies, be tampered by a sacrificial water layer[2]. Reproduced from Yoon et al., Scientific Reports **6**, 24791 (2016) under the Creative Commons Attribution 4.0 International License.

(ionization and atomic displacement), were then fed into computational reconstruction using the Expand–Maximize–Compress (EMC) algorithm for orientation and the Difference Map (DM) algorithm to solve the phase problem. The resulting 3D electron density maps could then be compared to the PDB model that was used as a sample specification in the simulation. The structure in the PDB had been measured with the NMR method. We reproduce in Fig. 1 the corresponding results from the original article by Yoon et al. [3]. The results demonstrate the predictive power of our start–to–end simulation toolchain. All main features of the protein were reproduced. Regions close to the surface show enhanced displacement and poorer agreement with the experimental data due to radiation damage.

1.2. Coherent Nanocrystal diffraction

Two codes integrated in *simex_platform* can be used to model x–ray diffraction from crystalline samples. The first code is *singfel*, which calculates the diffraction from given atomic coordinates and scattering formfactors. The second code is *pattern_sim* (part of *crystfel*) [4], which reads only the crystallographic unit cell and form factors plus information about the point group and sample size

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

to calculate the diffraction pattern consisting of Bragg spots and incoherent contributions. *singfel* reads the structure of the entire nanocrystal, thus providing more flexibility to account for radiation damage effects which break the translational symmetry of the sample. We validate both codes

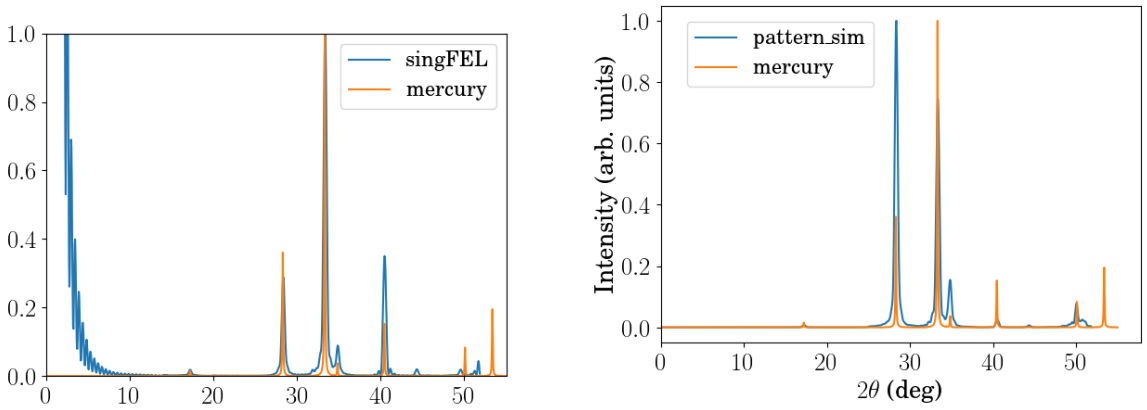


Figure 2: Simulated powder diffraction patterns (blue lines) for nanocrystalline fcc Fe_2O_3 using the diffraction modules *singfel* (left) and *pattern_sim* (right) compared to reference calculations for an infinite fcc lattice using the code *mercury* (orange curve).

against the code *mercury* [5] of the Cambridge Crystallographic Data Centre (CCDC). Fig. 2 shows the simulated powder diffraction pattern for *singFEL* vs. *mercury* (left) and for *pattern_sim* vs. *mercury* (right). The sample is a 18 nm sized fcc crystal of Fe_2O_3. In both cases, the positions of Bragg peaks produced by SIMEX correspond to the reference simulation. In general, the peak amplitudes differ between the SIMEX simulations and the reference model, while the *singFEL* results are in better agreement to the reference model than the *pattern_sim* results. This difference between *singFEL* and *pattern_sim* can be attributed to the more precise and well validated model for the form factors, calculated with the code *XATOM* [6]. Note that the *singFEL* calculation contains a large signal at small angles, this has been removed in the *pattern_sim* calculation. These simulations were used to support the feasibility of a experiment proposed at European XFEL.

In the following, we will apply the *singFEL* module for diffraction simulations and compare to experimental data. Fig. 3 compares *singfel* simulations to diffraction data from an experiment at the CXI endstation at LCLS in a virtual powder representation. The sample is fcc nanocrystalline C_{60}. Bragg peaks were identified in each measured 2D diffraction pattern using the *psana* [7] peak finder and histogrammed according to the peak's radial distance from the geometrical center of the detector image corresponding to zero scattering angle. The histogram (orange bars) is plotted as a function of the Bragg angle 2θ and compared to a SIMEX simulation using *singfel* (blue curve) and the *mercury* reference calculation for a perfect infinite fcc lattice. Similar to the model–vs–model comparison, the agreement between models and data is overall satisfying.

1.3. Imaging of High–Power Laser driven targets

In Deliverable Report D4.3 [9], we presented 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) images using the scattering code ParaTAXIS. A Particle In

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

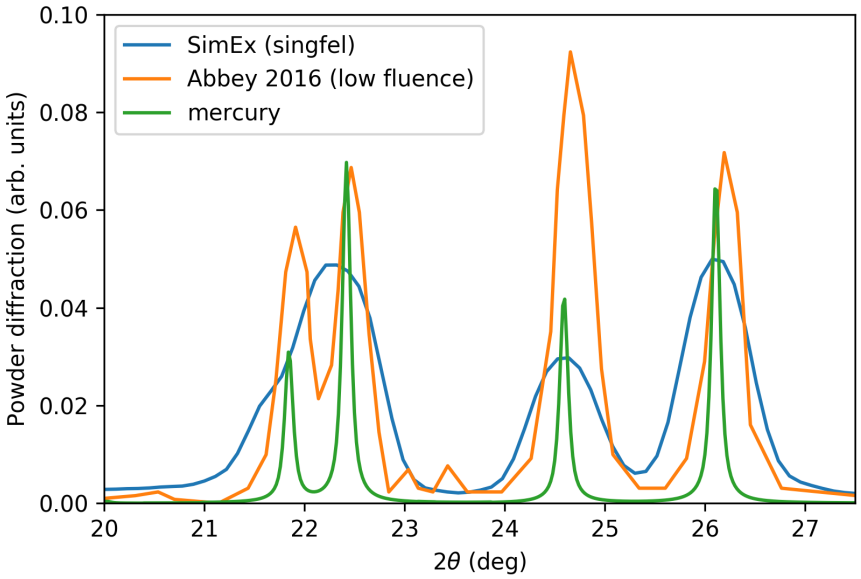


Figure 3: Virtual powder pattern from nanocrystalline C_{60} [8] (orange curve) compared to SIMEX simulations for a 10x10x10 C_{60} fcc supercell using the code *pysingfel* (blue curve) and compared to a modeled powder pattern for an infinite C_{60} fcc lattice using the code *mercury*.

Cell (PIC) simulation provides the time evolution of the electron density. XFEL Photons than scatter from this electron density. We assumed invariance of the target in the propagation direction and simulated the XFEL pulse with 10^{12} photons for which the target is optically thin. The ion density follows the electron density as expected for plasma expansion into vacuum. 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. The full details were reported in Deliverable D4.3 [9] and Deliverable D4.4 [10]. The simulation parameters match an experiment led by the group of HZDR which was recently published in [11]. In the experiment a silicon grating was irradiated by a Ti:Sa laser with 400 mJ on target and pulse duration 80 fs, focus size 1500 μm^2. The expansion of the grating as a function of time was measured and is reported in [11]. Here we compare the scattering patterns at $t = 0$ (i.e. when the laser maximum has reached the target front surface) between the ParaTAXIS simulation, an analytic solution and the measurement (see Fig. 4). For optically thin targets, the scattering pattern of a grating with soft edges can be described by an analytic equation ([11]). As an input into this equation we extract the sharpness of the grating edges from the simulation, $\sigma \approx 8$ nm We find a good agreement between the simulation and the analytic solution, as well as with the experimental curve when we take into account a few key aspects. First, we note that the ParaTAXIS simulation gives the same relative peak heights as the Fourier transform, validating the simulation. The signal level between the main peaks is different however due to the finite number of quasi-photons used in the simulation. With more simulation time, this level will further decrease, as this contribution is solely due to limited sampling of the full parameter space for the quasi-photons and will be fully extinct in the limit of high photon numbers. Secondly, the experiment contains signal from a background contribution, e.g. from plasma self-emission, bremsstrahlung and 3rd harmonic of the drive laser. Also, the experiment shows peaks

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

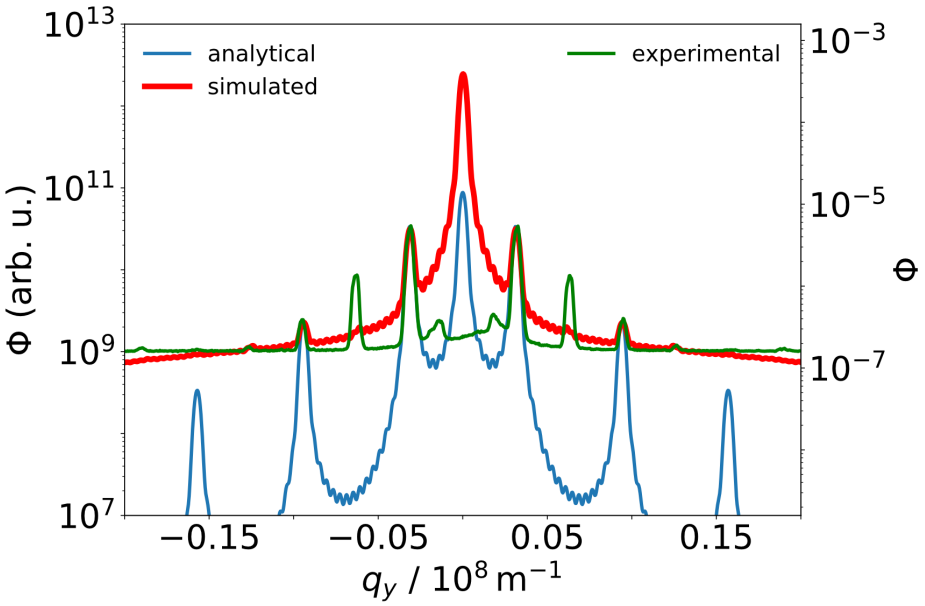


Figure 4: Comparison between the ParaTAXIS simulated scattering profile (red) and the analytic solution (blue), both based on PIC-simulated data used also in [Kluge et al., "Observation of Ultrafast Solid-Density Plasma Dynamics Using Femtosecond X-Ray Pulses from a Free-Electron Laser", Phys. Rev. X **8** (2018).]. The two curves almost perfectly agree in the relative heights of the scattered peaks, the signal in between is larger in the ParaTAXIS simulation due to the finite number of quasi-photons in the simulation. The simulation also agrees reasonably well with the experimental data, taking into account the finite experimental background and imperfect peak-to-valley aspect ratio of the front side grating, generating imperfect suppression of even scattering peaks.

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

in between the simulated peak positions, which are due to non-perfect aspect ratio of the pre-inscribed grating on the target surface. In the simulated case, the initial peak-to-valley aspect ratio is 1:1, while in the experiment due to production uncertainties this is not exactly the case. Besides this difference the experimental results are reproduced well by the simulation.

1.4. Warm dense matter x–ray absorption spectroscopy

Interaction of high–energy optical laser pulses with solid targets and shock compression is modeled with radiation–hydrodynamics (RH) simulations using either 1D (*Esther* [12]) or 2D (*Multi2D* [13]) RH implementations. The resulting profiles for mass density, temperature, ionization, and pressure are then fed into an electronic structure calculation. Finally, a real–space Green function code (e.g. FEFF [14] calculates the x–ray absorption spectrum. At this point, the spectrum and intensity distribution of the x–ray probe beam, delivered by x–ray raytracing simulations can be taken into account. The workflow, supported by SIMEX is shown in Fig. 5. Fig. 6 shows details on

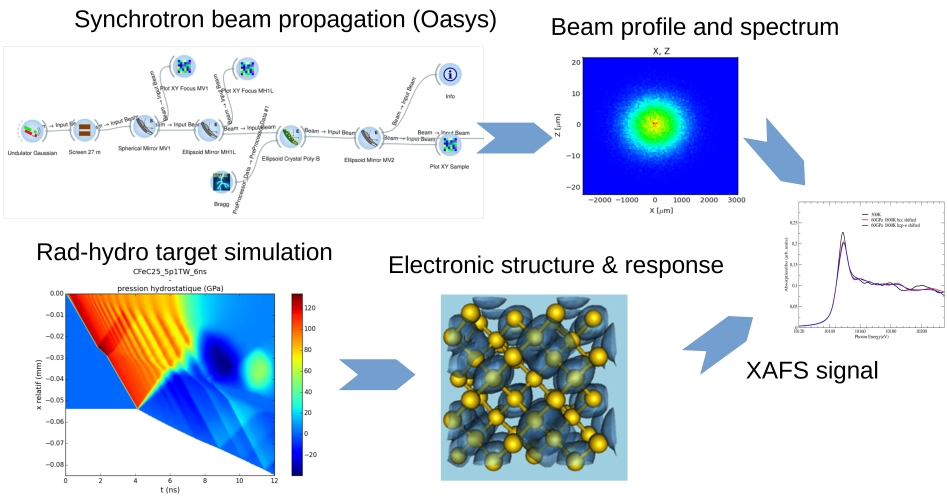


Figure 5: SIMEX workflow for simulations of x–ray absorption spectroscopy of shock–compressed warm dense matter generated by high–energy long pulse laser –matter interaction.

the experiment carried out at beamline ID24 at ESRF against which we validate our simulations. In Fig. 7, we show experimental EXAFS spectra, measured at beamline ID24 at ESRF [15] compared to ab–initio simulations assuming pressure and temperature conditions similar to those of the experiment, which were modelled with RH simulations. The experimental spectra measured at 500 GPa pressure and 1.7×10^4 K temperature (upper left) and at 120 GPa pressure and 2.7×10^3 K temperature (lower left), respectively, are shown in together with spectra measured at ambient conditions (black curve). The labels a,b,c, and d mark points in the spectrum, where the strongly compressed case shows significant differences with respect to the ambient case. At increased pressure, a shift

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

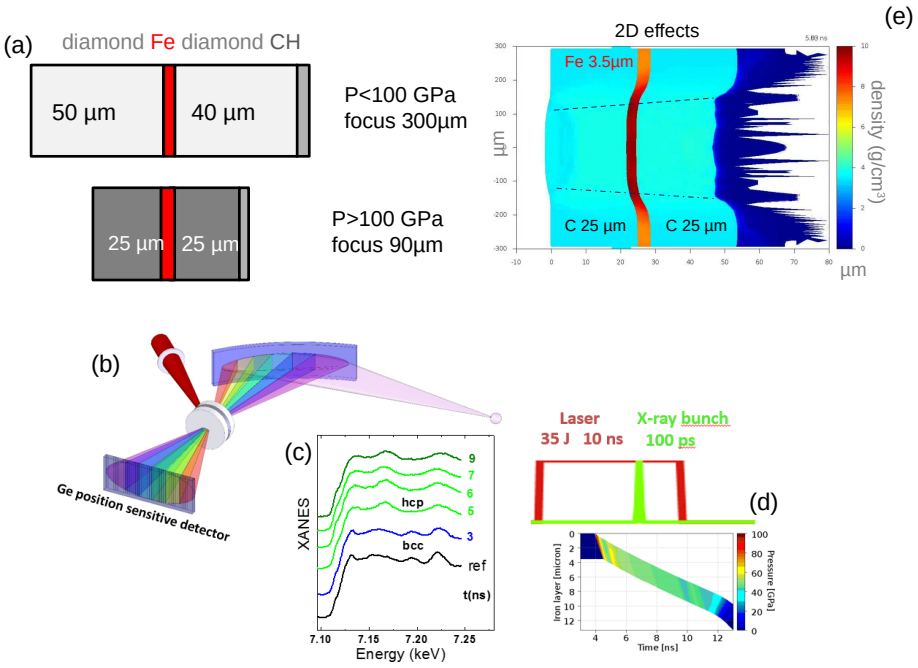


Figure 6: Target geometry (a), experimental setup (b), and XANES spectra (c) from ESRF ID24 experiment as reported in [Torchio et al. Scientific Reports **6**, 26402 (2016)], 1D (d) and 2D (e) radiation–hydrodynamics simulation of the laser–matter interaction.

of the K–edge onset to smaller energie is found, as well as a steepening of the edge profile in the lower half of the edge retion (<7.12 keV), whereas the absorption rises less steeply with respect to the ambient case in the upper half of the edge (>7.12 keV). The modelled spectra reproduce these differences qualitatively. A more quantitative analysis and comparison beteen theory and experiment is still outstanding. We also compared 1D RH simulations for the pressure as a function of energy on target in the case of molybdenum. Fig. 8 shows three measured datasets (dots) differing the the thickness of the phase plate used to smoothen the intensity distribution over the focal spot and the pump pulse duration. The solid lines mark the RH simulations, systematically overestimating the pressure. This is a typical artifact of 1D RH simulations. The systematic offset between model and data could be used to define a heuristic correction factor to apply to 1D simulation data.

1.5. X–ray diffraction diagnostics of coherent acoustic waves

The structural dynamics and the according X-ray diffraction response in one-dimensional sample structures after ultrafast laser excitation is modeled with *udkm1Dsim toolbx* [16] [https://www.github.com/eucall-software/udkm1Dsim]. First the acoustic response of a nanometric bilayer sample is simulated and then fed into a dynamical x–ray diffraction simulation. The resulting time-dependent diffraction curves are compared with experiments carried out at ID09 of the ESRF. Fig. 9 a) shows the experimental setup of the ID9 beamline. Pump-probe experiments with a single x-ray pulse of 100 ps duration are performed with a 1 kHz repetition rate. In addition, a sketch of the

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

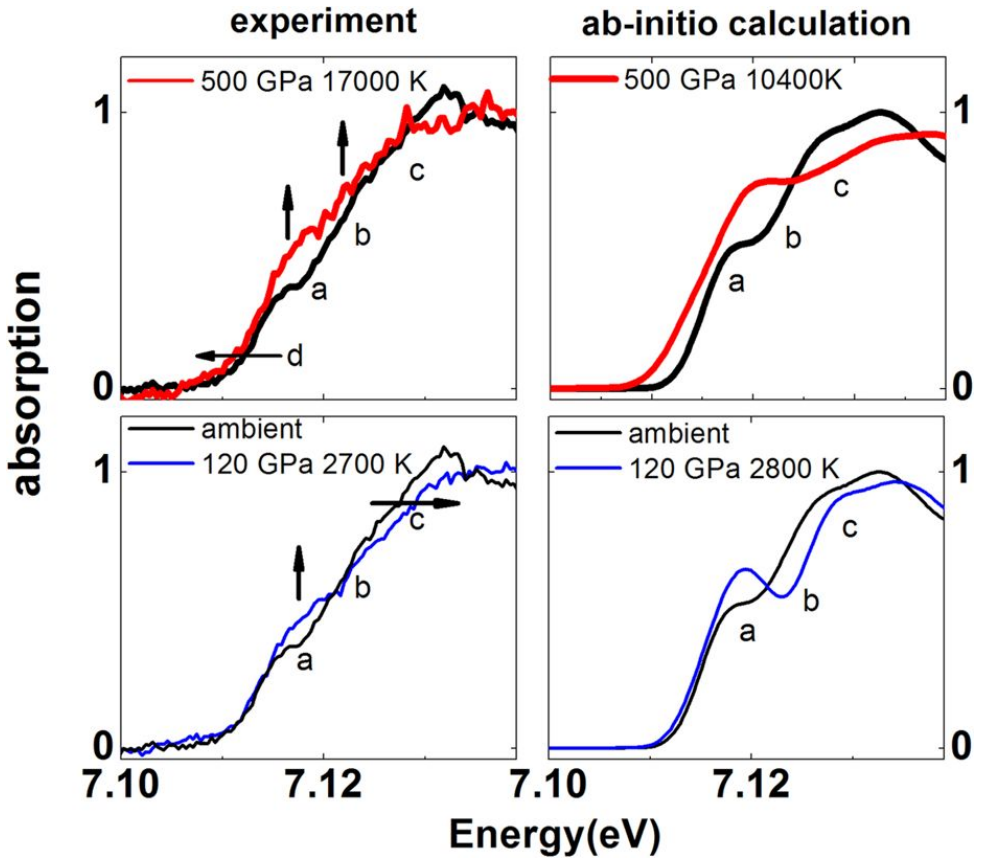


Figure 7: Left: Experimental EXAFS spectra close to the K–edge at two different pressure and temperature conditions realized in the experiment compared to ambient conditions (black line). Right: ab-initio molecular dynamic simulations. Labels a,b,c,d mark points in the spectrum where strong deviations from the ambient spectrum is observed as indicated by black arrows. These are qualitatively reproduced in the ab–initio simulations. Figure reproduced from [Torchio et al. Scientific Reports **6**, 26402 (2016)] under Creative Commons Attribution 4.0 International License.

bilayer sample structure is presented in Fig. 9 b). The sample is composed of 100 nm $LaAlO_3$ (LAO) and 50 nm $LaSrMnO_3$ (LSMO). Fig. 9 c) shows a simulation of the time-dependent strain profile of the nanometric bilayer sample after ultrashort optical excitation. The simulation includes the generation and propagation of coherent acoustic sound waves and covers the heat generated strain as well. Fig. 9 d) presents x-ray diffraction simulation of the excited sample for different pump-probe delays.

A benchmark experiment was performed to compare experimental results with the structural dynamics simulations. We extract the propagation time of the coherent acoustic sound wave, the strain amplitude of the sound wave and the time-dependent diffraction intensity of LAO and LSMO layers. The results are shown in Fig. 10.

The simulations qualitatively reproduce the structural dynamics of the laser excited sample. The measured 30 ps oscillation of the LAO (002) Bragg peak is simulated correctly. The sound wave is reflected at the free sample surface and enters the optical excited LSMO after 32 ps which leads to an additional expansion and successive to a shift of the LSMO (002) bragg peak to lower diffraction

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

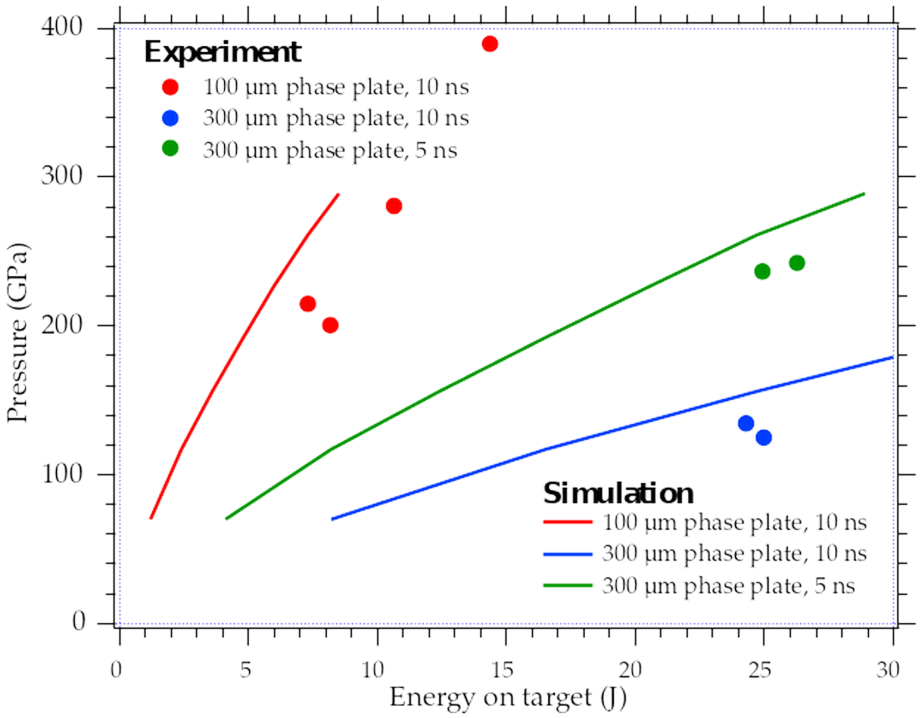


Figure 8: Pressure vs. energy on target (molybdenum) for three different phase plate thicknesses and pulse durations. The 1D radiation–hydrodynamics simulations (solid lines) systematically overestimate the experimental conditions (dots).

angles. We extract the amplitude of the coherent acoustic sound wave from the intensity and the Bragg peak position and find an excellent agreement between experiment and simulation.

1.6. Wavefront propagation through phase gratings

simex_platform interfaces the the coherent wavefront propagation code library *WPG* [https://www.github.com/samoylv/WPG]. In order to simulate wavefront measurement experiments using phase grating techniques [17], we have extended the set of optical elements supported by *WPG* by a selection of various 2D transmission gratings. The Grating element is based on the *srwl* class SRWLOptT (Optical Element: Transmission - generic) and its transmission properties are achieved by modifying attribute arTr - amplitude transmission and optical path difference as function of transverse position.

Grating structure can either be chosen from a predefined set (Fig. 12) or custom-made using a limited set of parameters. The predefined shapes are configured by pitch in the direction of x-axis and the y dimension can be scaled accordingly or be intentionally stretched to create rectangular structures.

The range of parameters for custom-made gratings include (Fig. 13)

1. vectors $\vec{a}$ and $\vec{b}$ to define spacing of structural elements
2. switch to enable halfway combinations (mesh to checkerboard)
3. fractional parameter for $\vec{a}$ and $\vec{b}$ to specify element dimensions

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

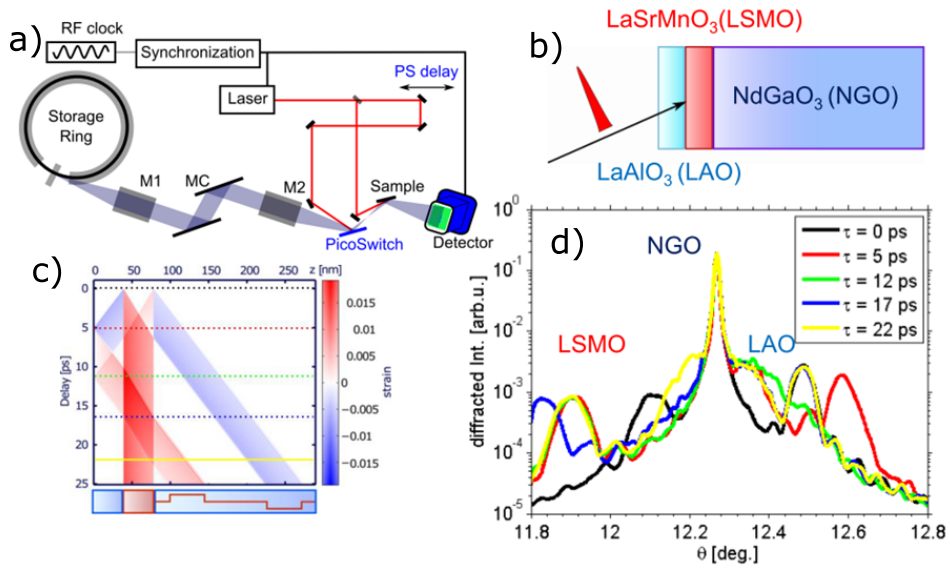


Figure 9: X–ray diffraction measurements of coherent acoustic waves: Schematic of the experiment (a), Sample structure and probe geometrt (b), strain as function of pump–probe delay and sample coordinate (c), and diffraction signals as function of pump–probe delay.

4. switch to turn element shapes into circles (hex)
5. roll grating along the beam

1.6.1. Grating simulation

We run the sample "S1 SPB CRL simplified beamline" from the WPG package(Fig. 14) with a 4 µm pitch, $\pi/2$ phase checkerboard grating being added 3 m downstream of focus and wavefront measured at distances between 5.625 m and 5.84 m from focus with 5 mm steps. Due to the Talbot effect from spherical wavefront, a variation in peak visibility in Fourier spectrum is to be expected in this range, which we evaluated. At chosen "detector" range, one full period of Talbot order (even-odd-even) occurs for the diagonal pitch and two periods occur for orthogonal pitch (along x/y axis), following the equation 1.

$$n(D) = \frac{2 * \lambda * \mu^2 * (D - G) * G}{D * pitch^2} \tag{1}$$

Where λ is wavelength, μ specifies $\pi/2$ (1) or π (2) phase shift, pitch is the grating pitch and D and G are distances from focus to detector and grating.

Visibility should be high for odd and low for even Talbot orders. A way to visualize it is by quantifying proximity to odd positions

$$oddness(x) = 1 - (x \,\%\, 2 - 1)^2$$

The expected visibility (Fig. 15, left) is given by equation 2. Since diagonal direction is usually dominant, orthogonal visibility is shown with factor of 0.5.

$$visib(D) = oddness\,(n(D)) \tag{2}$$

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

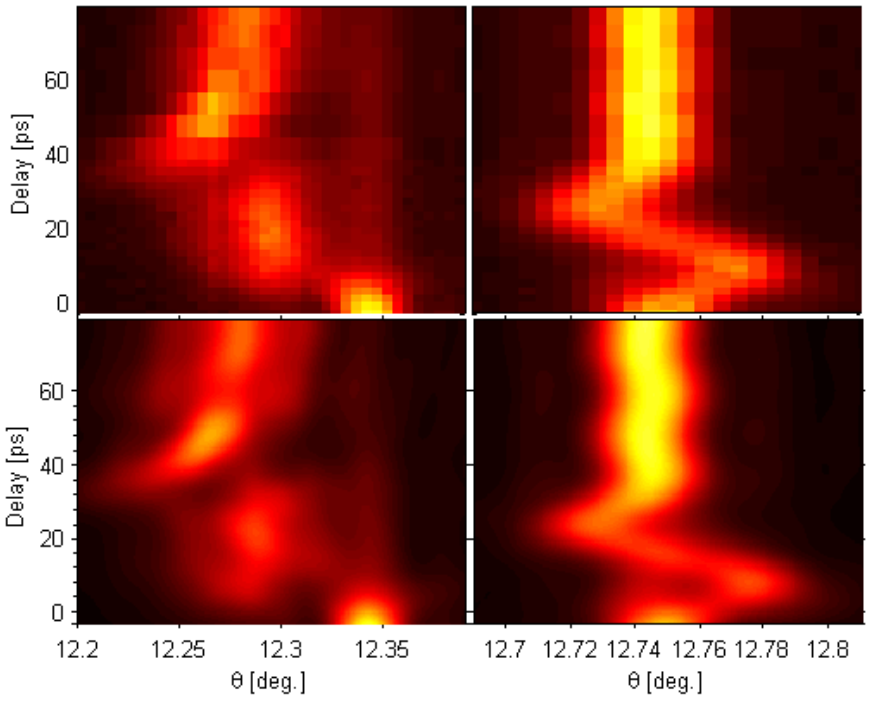


Figure 10: Time-dependent diffraction efficiency of the nanometric LAO and LSMO layers. The two upper panel show the measurement of the (002) Bragg peak position and intensity of LAO and LSMO as a function of the diffraction angle θ and pump-probe delay. The two lower panels present the simulated position and diffraction intensity of the (002) bragg peaks of LAO and LSMO [16].

Values calculated from the simulated beamline (Fig. 15, right) closely resemble the expected pattern and agree with other results we got from actual experiments. We observed a slight jitter not only in simulation visibility values, but also directly in wavefront calculated intensity. All the simulations were run with parameter *num_points* set to 1000 and the grating defined at 2000 data points (20 per orthogonal pitch). More precise results could be expected at higher resolutions at the cost of increased computation times.

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

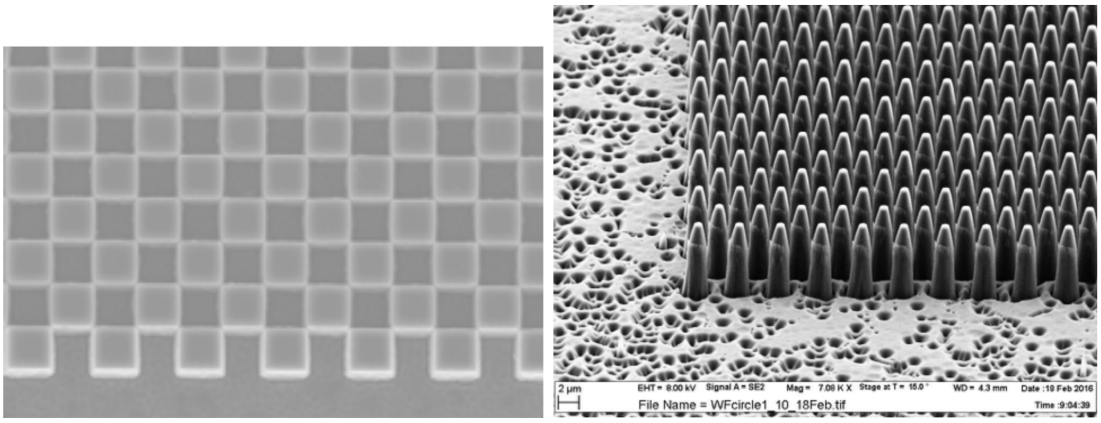


Figure 11: Checkerboard and hexagonal gratings used in conducted experiments

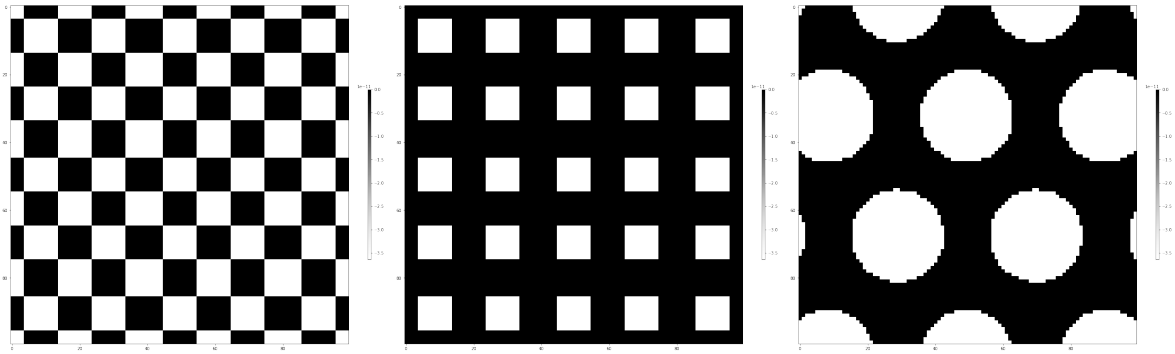


Figure 12: Predefined grating structures: checkerboard, mesh and hexagonal

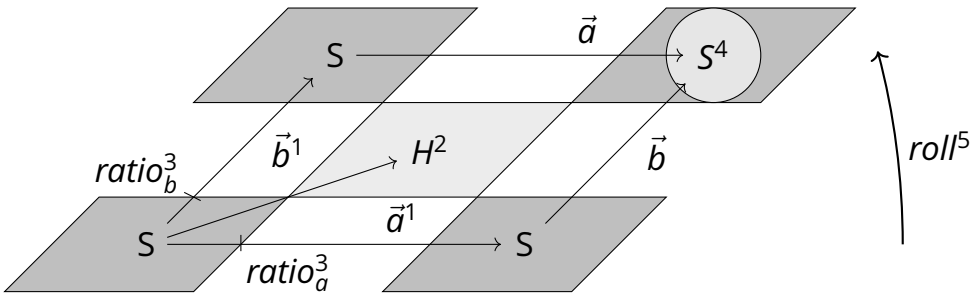


Figure 13: Custom grating parameters

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

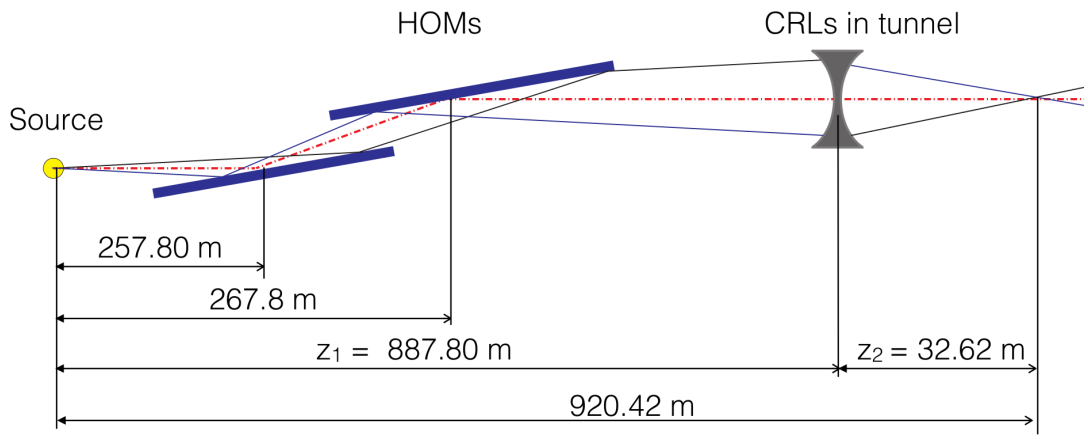


Figure 14: Simulated simplified beamline.

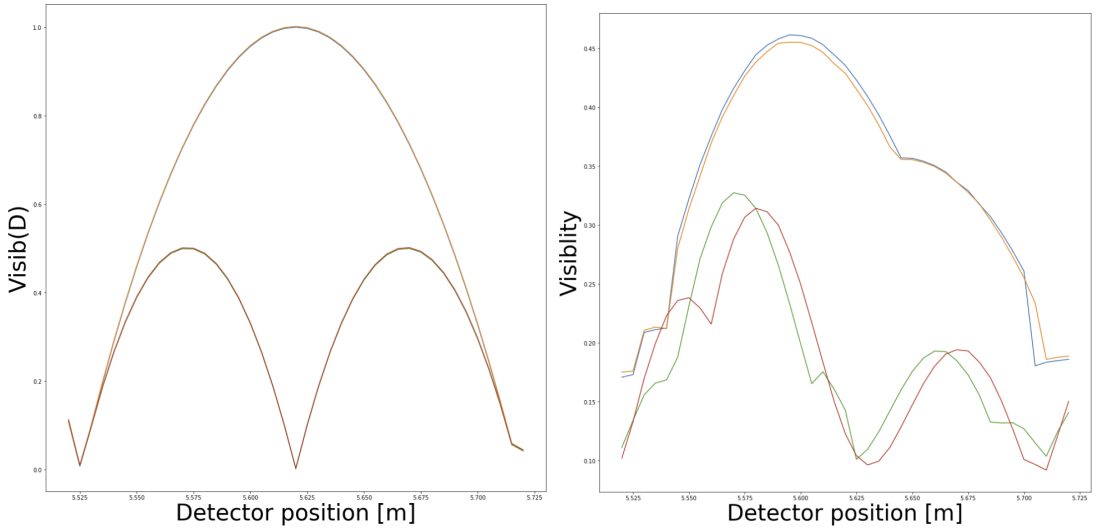


Figure 15: Visibility: expected and simulated. Diagonals: orange/blue; x/y: green/red

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

2. High Performance Computing Benchmarks

2.1. Single–particle imaging simulation pipeline

The single–particle imaging simulation pipeline consists of

1. pulse propagation
2. photon–matter interaction
3. diffraction
4. detector response (optional)

Each of these modules have very different and demanding computational resource requirements. We present benchmarking results for few selected modules which represent performance bottlenecks in the simulation pipeline.

2.1.1. Wavefront propagation

The wavefront propagation utilizes the software Synchrotron Radiation Workshop (SRW), a legacy C library with historically very limited options for parallelization. Recently, there have been three major developments:

1. Insertion of openMP macros to enable shared–memory parallelism
2. Multicore parallel execution of propagation of multiple FEL source pulses.
3. MPI parallel calculation of independent coherent modes for partially coherent wavefront propagation.

1) and 2) have been realized in SIMEX, 3) is an independent development in the radiation source simulation and propagation framework SiRepo `www.github.com/radiasoft/sirepo`. Fig. 16 shows the speedup of a single SRW wavefront propagation resulting from the insertion of openMP pragmas in the SRW source code.

2.1.2. Radiation damage

The radiation damage module is by far the most compute intensive part of the SPI pipeline, determining $\approx$ 90% of the total wall time [18] of one start–to–end propagation. The code *xmdyn and xatom*, provided by CFEL, DESY [19] utilizes GPGPU cards. MPI or shared memory parallelism are

Threads x MPI processes	Number of nodes	Wall time (min)	Time/input file (s)
1x1	1	660	1031
40x1	1	65	98
4x10	4	7.5	45
8x5	8	4.2	51

Table 1: Total wall time taken by the propagation of 40 individual source files with hybrid openMP/MPI parallelism in SRW. Note how the distribution of MPI processes over nodes and threads influences the walltime and the time to process each file.

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

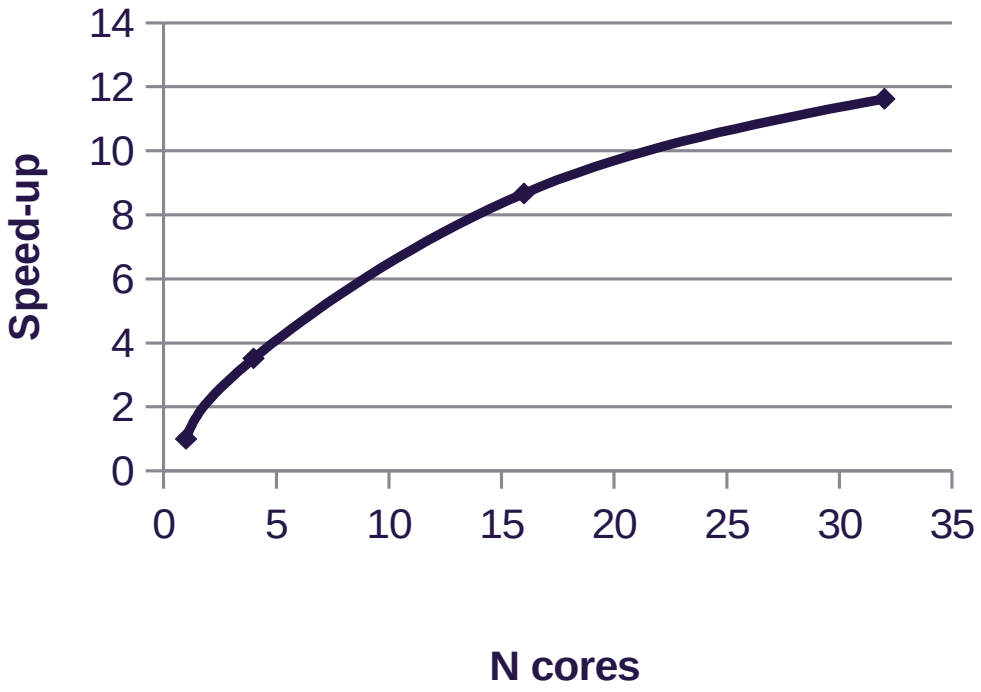


Figure 16: Speedup of WPG/SRW wavefront propagation for a single source input file using openMP shared–memory parallelism.

GPU model	Total wall time first run (s)	Total wall time subseq. run (s)
K20X	580	568
K40X	484	470
P100	470	466

Table 2: Wall time for a single *xmdyn and xatom* run on different NVIDIA(C) GPGPU cards. The first run takes longer because the atomic transition database has to be populated.

currently not implemented, only one GPGPU card can be utilized at a time. Our benchmarks therefore compare the performance on different GPGPU variants. The test case is a trajectory of the 2NIP protein irradiated by a 10 fs (FWHM) pulse of 5 keV FEL photons (pulse energy ≈1 mJ).

2.1.3. Diffraction

Diffraction calculations are performed with the code *pysingfel*. To enhance performance, this code uses the *numba* library to facilitate just–in–time compilation of compute intensive parts on the available hardware.

Typically, one simulation run produces many diffraction patterns. Since all pattern calculations are independent, the best speedup can be obtained by parallelizing the diffraction simulation over the number of diffraction patterns requested. Fig. 17 shows the speedup by comparing the Wall times for computation of 100 patterns on a shared memory system with 1, 10, and 100 MPI processes, respectively. The case with 100 MPI ran on 2 separate nodes, whereas the cases with 1 and 10 processes ran on a single node with shared memory. The additional data communication between the nodes results in a reduced speedup between 10 and 100 nodes as compared to the

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

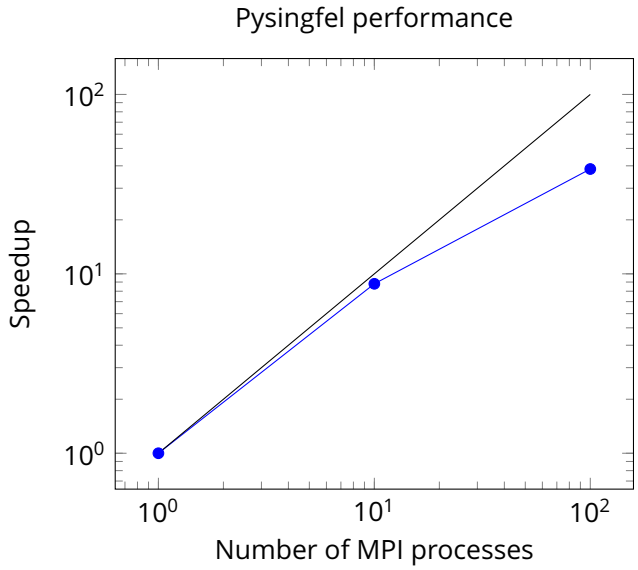


Figure 17: Speed up factor for parallel computation of 100 diffraction patterns with *pysingfel* using MPI parallelization. (the black line represents linear scaling).

speedup between 1 and 10 nodes. Also, the serial portion of the code (parsing the sample file, geometry calculations) result in a sub–linear speedup.

2.2. High–power laser–matter interaction (*picongpu*)

Performance benchmarks for the particle–in–cell code *picongpu* for the simulation of high–power laser–matter interaction are presented in [20], see also Ref. [21] for an updated version. We show in Fig. 18 the speedup (strong scaling) achieved by increasing the number of GPUs between 2 and 1024 demonstrating the near–linear scaling. More performance analysis data is presented in Ref. [20] including scalings up to 18432 GPUs and also weak scalings.

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

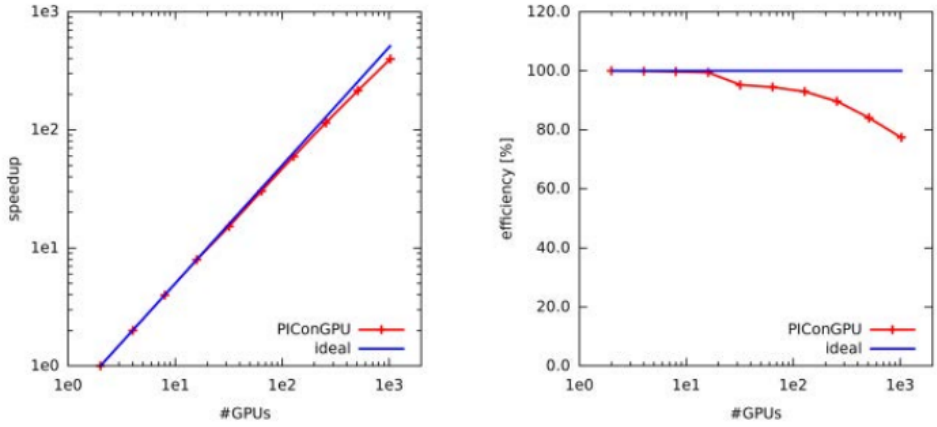


Figure 18: Speedup (left) and efficiency (right) from strong scaling performance analysis between 2 and 1024 GPUs for the PIC code *picongpu*. Reproduced from [Bussmann et al. "Radiative signatures of the relativistic Kelvin-Helmholtz instability", Proc. SC'13 (2013)].

3. SIMEX as a Service

3.1. Introduction

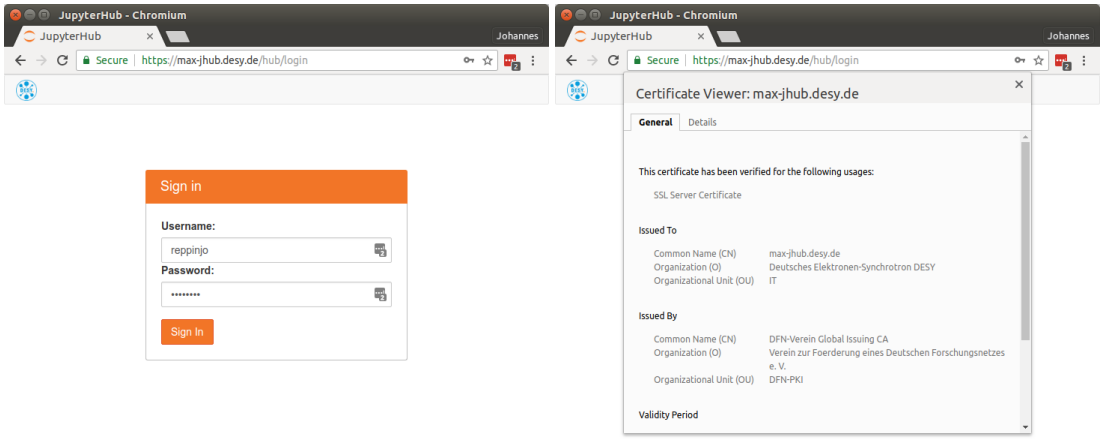
For the task 4.3.2 *Provide a framework towards a service to users [...]* we set up an instance of *JupyterHub* for the users of the *Maxwell Cluster* at DESY. Users of the *SIMEX* code are familiar with the computing environment at DESY and use the cluster in many ways. To run heavy computations on the cluster users have to submit a job to *SLURM* which will then allocate ressources to the users' jobs and run them. This is a well-established workflow for simulations and data reduction pipelines where the result of the computation can be delayed by days from the time of submission. For other use-cases though, users need near to instantaneous feedback from their input. In this case, interactive compute environments like matlab(C), mathematica (C), root(C), or (i)python (C) allow a more direct interaction with the data. Here, individual commands or command blocks are evaluated on the fly and results can be displayed or further processed in the same environment. We built our solution on the opensource and widely distributed ipython interactieve python shell (backend) and the Jupyter Notebook as a frontend. The JupyterHub administers user credentials, resource configuration, and access to data and notebooks. It supports usage of the same computer cluster or even the same notebook by many users.

We developed example notebooks which can be used as a starting point by *simex* users to learn the workflow and to run their own simulations by modifying the example. We document in appendix A two example workflows for simulation of single–particle imaging and for nanocrystalography. Both notebooks can be obtained from the EUCALL software repository. `https://www.github.com/eucall-software/simex_notebooks`.

3.2. Features of the JupyterHub at DESY

The JupyterHub we set up uses a lot of infrastructure already available at DESY. In the following we explain how it was included into an instance to make data analysis more convenient. Features of the *JupyterHub* Instance include:

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



(a) View of the login Page of the JupyterHub instance https://max-jhub.desy.de at DESY

(b) The website has a trusted SSL certificate so users can trust our website.

Figure 19: Screenshots of the JupyterHub login page and its SSL certificate

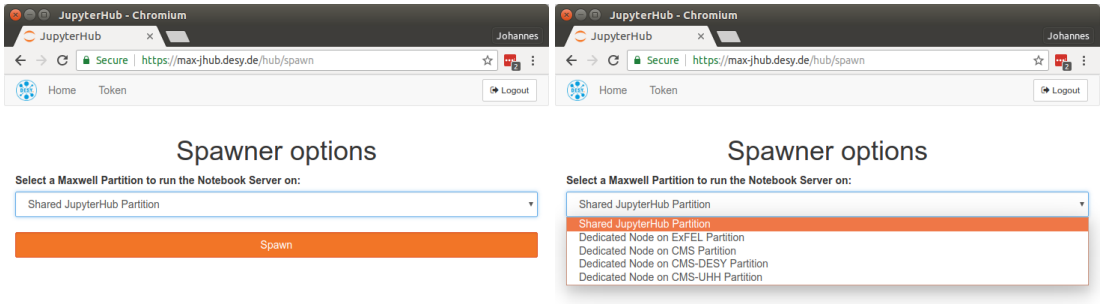
- Encryption via SSL using a DFN (Deutsches Forschungsnetzwerk) inheritance certificate to provide user data security
- Integration into the existing LDAP authentication at DESY
- Using the Maxwell Cluster as an infrastructure to provide computing power
- Connection to the SLURM ressource manager on the Maxwell Cluster
- Customization of the Notebook Spawner to provide access to the GPFS file system
- Addition of Various Python Kernel Versions and other demanded languages such as R & bash kernels

3.3. Using the JupyterHub and SIMEX

To use the JupyterHub at DESY one has to visit the website https://max-jhub.desy.de while on the DESY-XFEL Network. Opening the ports to the Internet is work in progress and needs the server to be isolated in the demilitarized zone (DMZ) which is a different network section and has special firewall rules. Users then see the screen shown in Fig. 19 which has been customized to show the DESY Logo and shows the *Secure* Website lock so users know their data is protected. Users then have to enter their DESY credentials which are passed to the LDAP server for authentication.

After logging on, users are presented with a choice, shown in Fig. 20, where their notebook should run. There is a shared partition where users can do lightweight calculations and configuration of larger computations. If users want to do data analysis that requires multiple CPUs and large amounts of memory they can select a whole node of their inidividual partition. It is not given that the notebook server will be able to start up immediately, though, since SLURM might need time to allocate a node. The shared partition, on the other hand, is responsive and starts the notebook server within a few seconds. On the right panel of Fig. 20 the options to start are shown. This is neccessary so users have access to their group specific (ExFEL, CMS, and others) partition if they want to do interactive computations with a full node and not just sharing ressources. Once the

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



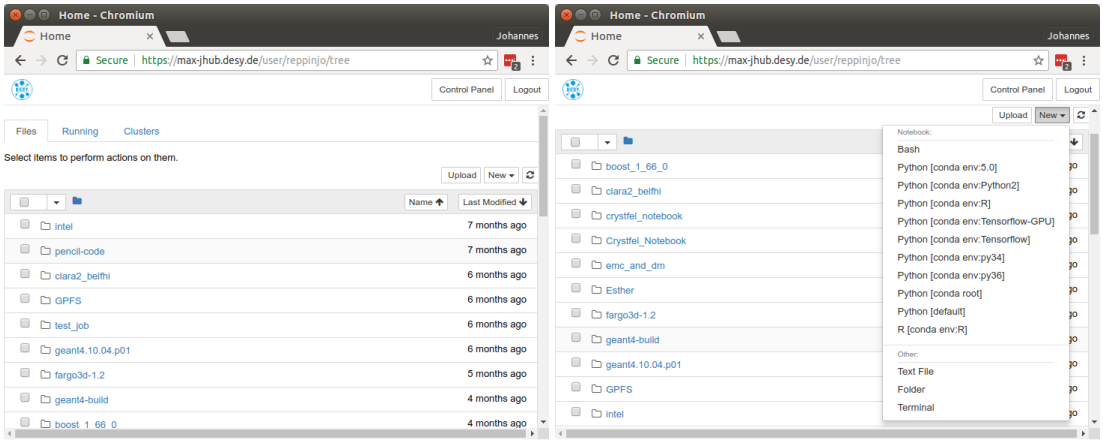
(a) Before getting to the Notebook, users see the choice of spawner options for the partition to run the notebook on.

(b) A shared Jhub partition and workgroup partitions for more heavy load usage are available, the shared one is the default.

Figure 20: After the login using the DESY credentials was successful, the user has multiple options to select from for the spawner to start the notebook server

user has selected the partition a SLURM job is submitted that starts the Notebook Server and a connection is established and users see the screen depicted in Fig. 21. Once the server is started and and a notebook is openend, users of *simex_platform* code can execute code cells to import the module and functions they use in their regular python-driven workflow they are used to without the Jupyter Notebook, as is shown in Fig. 22. The advantage is now that plots can be displayed inline, so no X-forwarding is needed. Also, the Jupyter Notebook supports cells with *Markdown* format, so documenting the work and coding progress can be done by adding cells with headings, text and even LaTeX-style text blocks.

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



(a) This is a the screen users see when the notebook is launched. The files and folders correspond to the users' home directorys.

(b) When choosing a new kernel users have many options of Python versions, R & a bash kernel, text files editing and a terminal session.

Figure 21: screenshots after a user has successfully logged onto the JupyterHub, the users see their files and folder as well as the available Kernels that can be used to start a Notebook.

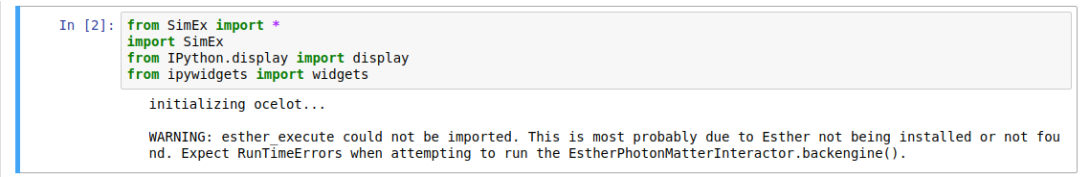


Figure 22: The Jupyter Notebook executes code in *cell blocks* that can be executed individually. The import statement to make the *SimEx* python modules available runs and user sees immediately if it fails or the import was successful

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

References

1. Schlessman, J. L., Woo, D., Joshua-Tor, L., Howard, J. B., and Rees, D. C., *Conformational variability in structures of the nitrogenase iron proteins from Azotobacter vinelandii and Clostridium pasteurianum*, *Journal of Molecular Biology* **280,** 669–685 (1998).
2. Hau-Riege, S. P., London, R. A., and Szoke, A., *Dynamics of biological molecules irradiated by short x-ray pulses*, *Physical Review E* **69,** 51906 (2004).
3. 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).
4. White, T. A., Kirian, R. A., Martin, A. V., Aquila, A., Nass, K., Barty, A., and Chapman, H. N., *CrystFEL: a software suite for snapshot serial crystallography*, *Journal of Applied Crystallography* **45,** 335–341 (2012).
5. Macrae, C. F., Bruno, I. J., Chisholm, J. A., Edgington, P. R., McCabe, P., Pidcock, E., Rodriguez-Monge, L., Taylor, R., van de Streek, J., and Wood, P. A., *Mercury CSD 2.0 – new features for the visualization and investigation of crystal structures*, *Journal of Applied Crystallography* **41,** 466–470 (2008).
6. Son, S.-K., Young, L., and Santra, R., *Impact of hollow-atom formation on coherent x-ray scattering at high intensity*, *Phys. Rev. A* **83,** 33402 (2011).
7. SLAC-LCLS, *The psana website*, https://confluence.slac.stanford.edu/display/PSDM/LCLS+Data+Analysis.
8. Abbey, B., Dilanian, R. A., Darmanin, C., Ryan, R. A., Putkunz, C. T., Martin, A. V., Wood, D., Streltsov, V., Jones, M. W. M., Gaffney, N., Hofmann, F., Williams, G. J., Boutet, S., Messerschmidt, M., Seibert, M. M., Williams, S., Curwood, E., Balaur, E., Peele, A. G., Nugent, K. A., and Quiney, H. M., *X-ray laser-induced electron dynamics observed by femtosecond diffraction from nanocrystals of Buckminsterfullerene*, *Science Advances* **2,** e1601186–e1601186 (2016).
9. 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* , 2017, https://dx.doi.org/10.5281/zenodo.998647.
10. Fortmann-Grote, C., Garten, M., Huebl, A., Grund, A., Kluge, T., Bussmann, M., and Mancuso, A. P., *Deliverable D4.4* , 2017, https://dx.doi.org/10.5281/zenodo.998644.
11. Kluge, T., Rödel, M., Metzkes-Ng, J., Pelka, A., Garcia, A. L., Prencipe, I., Rehwald, M., Nakatsutsumi, M., McBride, E. E., Schönherr, T., Garten, M., Hartley, N. J., Zacharias, M., Grenzer, J., Erbe, A., Georgiev, Y. M., Galtier, E., Nam, I., Lee, H. J., Glenzer, S., Bussmann, M., Gutt, C., Zeil, K., Rödel, C., Hübner, U., Schramm, U., and Cowan, T. E., *Observation of Ultrafast Solid-Density Plasma Dynamics Using Femtosecond X-Ray Pulses from a Free-Electron Laser*, *Physical Review X* **8**, doi:10.1103/physrevx.8.031068 (2018).
12. Colombier, J. P., Combis, P., Bonneau, F., Harzic, R. L., and Audouard, E., *Hydrodynamic simulations of metal ablation by femtosecond laser irradiation*, *Phys. Rev. B* **71,** 165406 (2005).
13. Ramis, R., Meyer-ter-Vehn, J., and Ramırez, J., *MULTI2D – a computer code for two-dimensional radiation hydrodynamics*, *Computer Physics Communications* **180,** 977–994 (2009).

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

14. Rehr, J. J., Kas, J. J., Prange, M. P., Sorini, A. P., Takimoto, Y., and Vila, F., *Ab initio theory and calculations of X-ray spectra*, *Comptes Rendus Physique* **10,** 548–559 (2009).

15. Torchio, R., Occelli, F., Mathon, O., Sollier, A., Lescoute, E., Videau, L., Vinci, T., Benuzzi-Mounaix, A., Headspith, J., Helsby, W., Bland, S., Eakins, D., Chapman, D., Pascarelli, S., and Loubeyre, P., *Probing local and electronic structure in Warm Dense Matter: single pulse synchrotron x-ray absorption spectroscopy on shocked Fe*, *Sci. Rep.* **6,** 26402 (2016).

16. Schick, D., Bojahr, A., Herzog, M., Shayduk, R., von Korff Schmising, C., and Bargheer, M., *udkm1Dsim – A simulation toolkit for 1D ultrafast dynamics in condensed matter*, *Computer Physics Communications* **185,** 651–660 (2014).

17. Weitkamp, T., Nöhammer, B., Diaz, A., David, C., and Ziegler, E., *X-ray wavefront analysis and optics characterization with a grating interferometer*, *Applied Physics Letters* **86,** 054101 (2005).

18. The wall time defines the time elapsed between the start and the end of a computer program run.

19. Jurek, Z., Son, S.-K., Ziaja, B., and Santra, R., *XMDYNandXATOM: versatile simulation tools for quantitative modeling of X-ray free-electron laser induced dynamics of matter*, *Journal of Applied Crystallography* **49,** 1048–1056 (2016).

20. Bussmann, M., Schmitt, F., Schramm, U., Schuchart, J., Widera, R., Burau, H., Cowan, T. E., Debus, A., Huebl, A., Juckeland, G., Kluge, T., Nagel, W. E., and Pausch, R., *Radiative signatures of the relativistic Kelvin-Helmholtz instability*, in *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis on - SC '13*, (ACM Press, New York, New York, USA, 2013), 1–12, http://doi.acm.org/10.1145/2503210.2504564%20http://dl.acm.org/citation.cfm?doid=2503210.2504564.

21. Zenker, E., Widera, R., Huebl, A., Juckeland, G., Knüpfer, A., Nagel, W. E., and Bussmann, M., *Performance-Portable Many-Core Plasma Simulations: Porting PIConGPU to OpenPower and Beyond*, *Lecture Notes in Computer Science* **9945,** 293–301 (2016).

A. Example workflows

The following pages contain jupyter notebooks converted to pdf output. The original notebooks can be obtained from the EUCALL software repository https://www.github.com/eucall-software/simex_notebooks.

A.1. Start–to–end single particle imaging

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

Header

In [4]:
```
# Activate matplotlib magic
%matplotlib inline
```

In [5]:
```
# Import all SimEx modules
import SimEx
from SimEx import *
```

```
/usr/lib64/python3.4/importlib/_bootstrap.py:321: FutureWarning: Conversion of the second argume
  return f(*args, **kwds)


initializing ocelot...
```

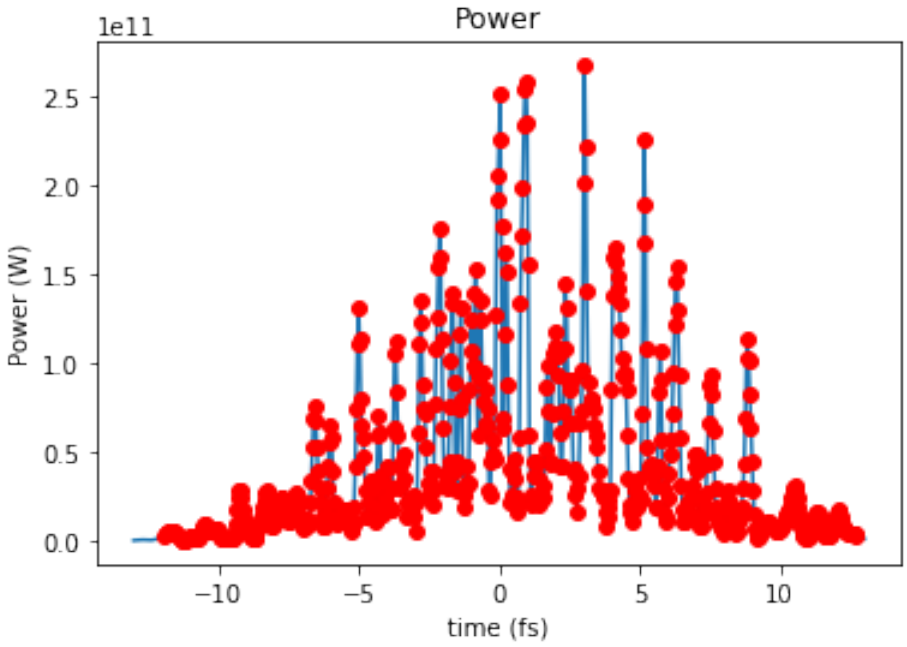
Step 1: Source diagnostics

In [6]:
```
source_analysis = XFELPhotonAnalysis(input_path="FELsource_out_0000001.h5")
```

```
 Start initialization.

 Loading wavefront from FELsource_out_0000001.h5.
 ... done.

 Getting intensities.
 ... done.
 Data dimensions =  (104, 104, 651)

 Masking NANs.
 ... done.
```

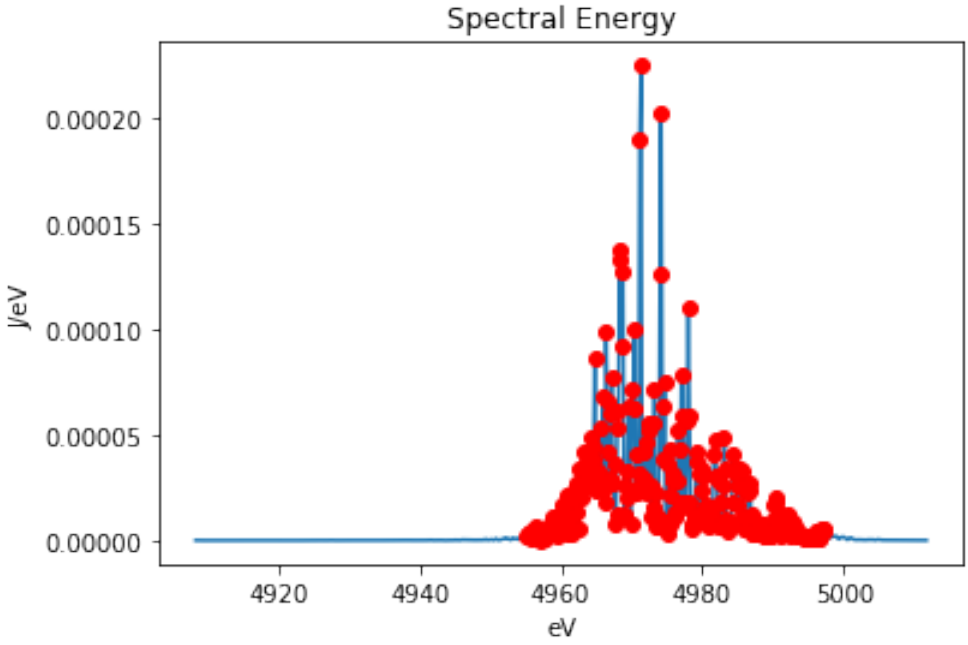
In [7]:
```
source_analysis.plotTotalPower()
```

```
 Plotting total power.
Pulse energy 0.001 J
```



In [8]: source_analysis.plotTotalPower(spectrum=True)

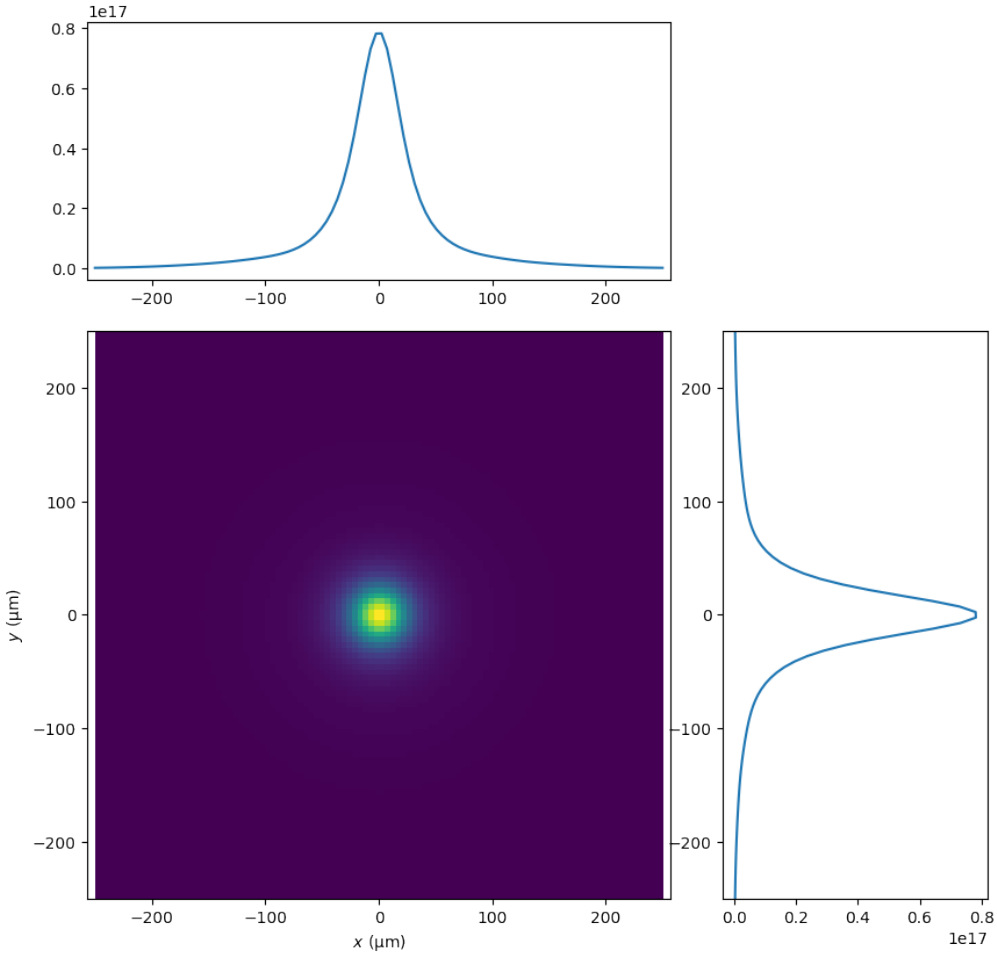
```
Plotting total power.

Switching to frequency domain.
```



In [9]: source_analysis.plotIntensityMap()

```
 Plotting intensity map.
R-space
```

```
<matplotlib.figure.Figure at 0x2b3d8e591cf8>
```



Step 2: Propagation through beamline

Import beamline for WPG

```
In [10]: from prop import exfel_spb_kb_beamline
```

Propagator setup

```
In [11]: propagation_parameters = WavePropagatorParameters(beamline=exfel_spb_kb_beamline)
```

```
In [12]: propagator = XFELPhotonPropagator(parameters=propagation_parameters,
                                           input_path='FELsource_out_0000001.h5',
                                           output_path='prop_out.h5')
```

```
In [ ]: #propagator.backengine()
```

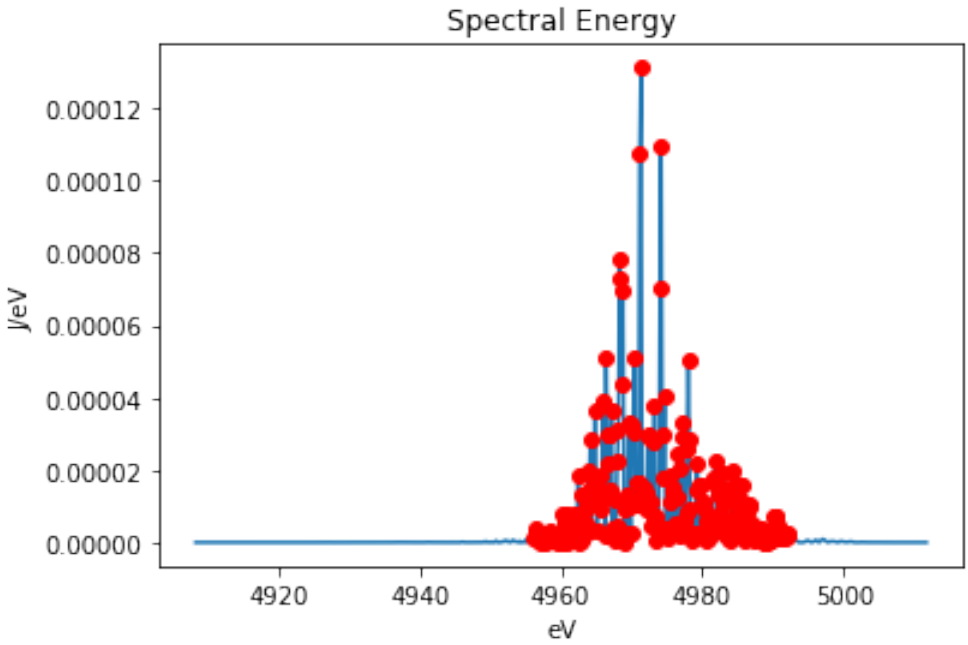
In [ ]: *#propagator.saveH5()*

In [13]: prop_analysis=XFELPhotonAnalysis(input_path='prop_out.h5')

```
Start initialization.

Loading wavefront from prop_out.h5.
... done.

Getting intensities.
... done.
Data dimensions =  (78, 78, 651)

Masking NANs.
... done.
```

In [14]: prop_analysis.plotTotalPower(spectrum=True)

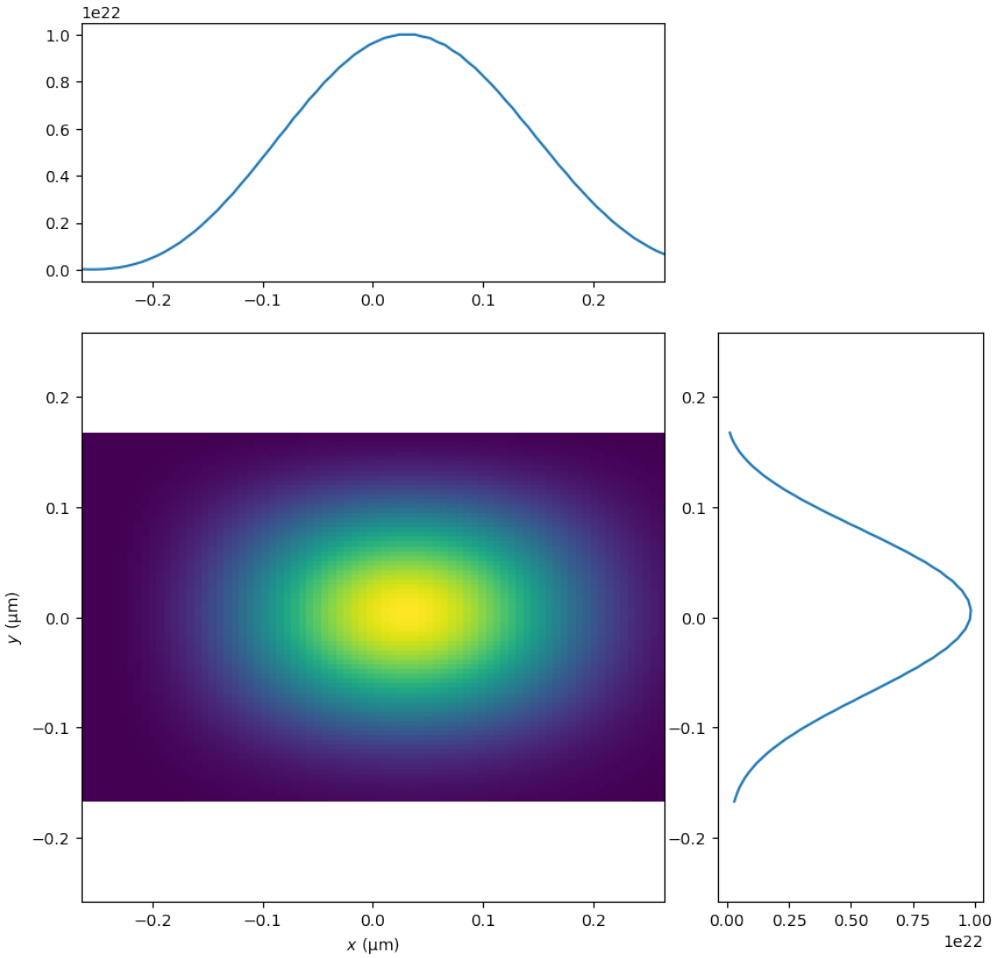
```
Plotting total power.

Switching to frequency domain.
```



```
In [15]: prop_analysis.plotIntensityMap()
```

```
 Plotting intensity map.
R-space
```

```
<matplotlib.figure.Figure at 0x2b3d8e5c5b70>
```



Step 3: Photon-Matter interaction (demonstrator)

```
In [16]: pmi_parameters={"number_of_trajectories" : 1,
                         "random_rotation" : False}
         photon_matter_interactor=XMDYNDemoPhotonMatterInteractor(parameters=pmi_parameters,
```

```
                                                    input_path='prop_out.h5',
                                                    output_path='pmi',
                                                    sample_path='5udc.pdb')
```

```
Sample file 5udc.pdb was not found. Will attempt to query from RCSB protein data bank.
```

```
In [17]: photon_matter_interactor.backengine(); photon_matter_interactor.saveH5()
```

```
Previous module:  s2e_spi
  NOT: data
  NOT: history
info
misc
params
version
['arrEhor', 'arrEver']
[('arrEhor', <HDF5 dataset "arrEhor": shape (78, 78, 651, 2), type "<f4">), ('arrEver', <HDF5 da
PDB file 5udc.pdb could not be found. Attempting to query from protein database server.
Downloading PDB structure '5udc'...
```

Step 4: Scattering

Configure Detector geometry

One panel

```
In [18]: panel = DetectorPanel(ranges={"fast_scan_min" : 0, "fast_scan_max" : 100,
                                       "slow_scan_min" : 0, "slow_scan_max" : 100},
                               pixel_size=6*220.0e-6*meter,
                               energy_response=1.0/electronvolt,
                               distance_from_interaction_plane=0.13*meter,
                               corners={"x" : -49, "y": -49},
                               saturation_adu=1.e6,
                               )
```

```
In [19]: detector_geometry = DetectorGeometry(panels=panel,)
```

Configure the Diffractor Parameters

```
In [20]: diffraction_parameters = SingFELPhotonDiffractorParameters(
                                                    uniform_rotation=False,
                                                    slice_interval=100,
                                                    number_of_slices=100,
                                                    number_of_diffraction_patterns=1,
                                                    detector_geometry=detector_geometry,
                                                    forced_mpi_command='mpirun -np 1',
                                                   )
```

Initialize the Diffractor

```
In [21]: diffractor = SingFELPhotonDiffractor(parameters=diffraction_parameters,
                                               input_path='pmi',
                                               output_path="diffr")
```

```
WARNING: Geometry not set, calculation will most probably fail.
```

Run the scattering simulation

```
In [22]: diffractor.backengine()
```

```
Out[22]: 0
```

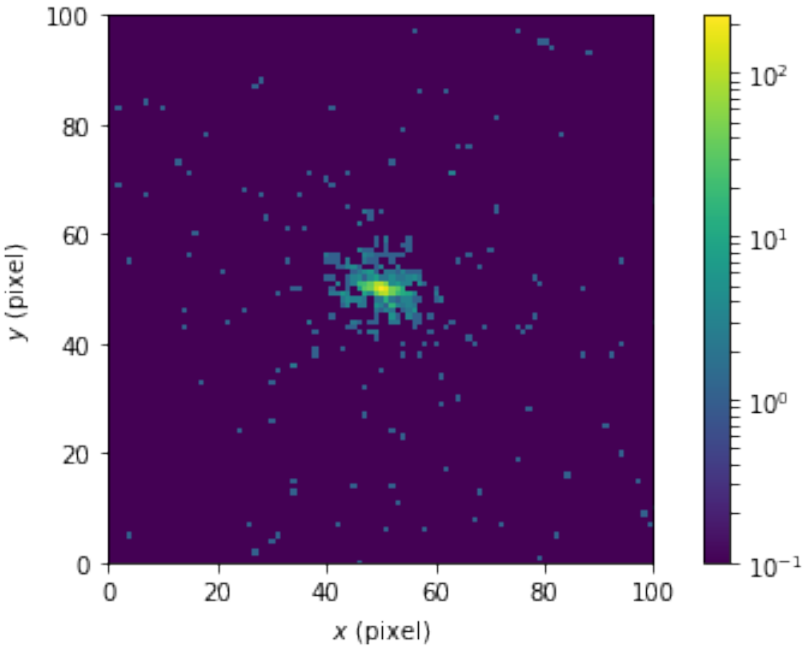
```
In [24]: diffractor.saveH5()
```

Analysis

Setup analysis object

```
In [25]: spi_analysis = DiffractionAnalysis(diffractor.output_path,
                                            pattern_indices=[1],
                                            poissonize=True)
```

```
In [26]: spi_analysis.plotPattern(logscale=True)
```



A.2. Nanocrystal diffraction

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

Header

```
In [1]: # Activate matplotlib magic
        %matplotlib inline
```

```
In [2]: # Import all SimEx modules
        from SimEx import *
```

```
initializing ocelot...
```

Configuration

Configure photon beam properties

```
In [3]: beam = PhotonBeamParameters(
                photon_energy = 5.0e3*electronvolt,
                beam_diameter_fwhm=2e-7*meter,
                pulse_energy=0.45e-3*joule,
                photon_energy_relative_bandwidth=0.003,
                divergence=0.0*radian,
                photon_energy_spectrum_type='tophat',
                )
```

Configure Detector geometry

One panel

```
In [4]: panel0 = DetectorPanel(pixel_size=8*200.0e-6*meter,
                               energy_response=1.0/electronvolt,
                               distance_from_interaction_plane=0.05*meter,
                               saturation_adu=1.e6,
                               corners={"x" : -63, "y": -63},
                               ranges={"fast_scan_min" : 0, "fast_scan_max" : 127,
                                       "slow_scan_min" : 0, "slow_scan_max" : 127},
                               )
```

```
In [5]: detector_geometry = DetectorGeometry(panels=panel0)
```

Configure the Diffractor Parameters

```
In [6]: diffractor_parameters = CrystFELPhotonDiffractorParameters(sample='5udc.pdb',
                                                                   crystal_size_range=[1e-7,1e-7
                                                                   number_of_diffraction_pattern
                                                                   beam_parameters=beam,
                                                                   detector_geometry=detector_ge
                                                                  )
```

```
PDB file 5udc.pdb could not be found. Attempting to query from protein database server.
Downloading PDB structure '5udc'...
```

Initialize the Diffractor

```
In [7]: diffractor = CrystFELPhotonDiffractor(parameters=diffractor_parameters,
                                               input_path=None,
                                               output_path="xstal_diffr")
```

Run the simulation

Launch simulation

```
In [8]: diffractor.backengine()
```

```
Out[8]: 0
```

Save data to hdf5

```
In [9]: diffractor.saveH5()
```

```
Renaming diffr_out_0000001.h5 to _0000001.h5.
```

Analysis

Setup analysis object

```
In [10]: analysis = DiffractionAnalysis(diffractor.output_path)
```

```
In [11]: analysis.plotPattern()
```