VINYL: The VIrtual Neutron and x-raY Laboratory and its applications

August, 2020

Authors: Juncheng E[1]

A. Hafner[2], T. Kluyver[1], M. Bertelsen[3], M. Upadhyay Kahaly[4], Z. Lecz[4], S. Nourbakhsh[5], A. P. Mancuso[1], C. Fortmann-Grote[1,6]

[1]European XFEL [2]CERIC-ERIC [3]European Spallation Source [4]ELI-HU

[5]Institut Laue-Langevin [6]Max Planck Institute for Evolutionary Biology

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 823852

Outline

- Introduction
- The architecture of VINYL
  - Backends
  - Interfacing data format
- Applications
  - Gaussian beam and its propagation
  - Serial crystallography simulation at European XFEL
  - Polychromatic diffraction simulation using OASYS and SimEx

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 823852

panosc
photon and neutron open science cloud

Findable Accessible Interoperable Reusable

Photon and Neutron Open Science Cloud (PaNOSC) is funded by EU Horizon 2020 until 2022 for:

- **Making data produced at Europe's major Photon and Neutron sources fully compatible with the FAIR principles**
- **Developing and providing services for scientific data (catalogue, analysis and simulation)**
- **Connecting these to the European Open Science Cloud (EOSC).**

CERIC Central European Research Infrastructure Consortium
eli delivery consortium
ESRF The European Synchrotron
European XFEL
ess EUROPEAN SPALLATION SOURCE
ILL NEUTRONS FOR SOCIETY
EGI
GÉANT

https://www.panosc.eu/

panosc

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 823852

WP5 Virtual Neutron and X-ray Laboratory

VIrtual Neutron and x-raY Laboratory (ViNYL) is a simulation environment to:

- **Offer services for simulation and modelling of photon and neutron instruments, as well as start-to-end simulations to describe entire experiments at photon and neutron facilities.**
- **Help estimate the most optimal experimental environment based on the final simulated data quality for a particular experiment.**
- **Make simulation data services inter-operable among partners and with data analysis services and data catalogues.**

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 823852

Start-to-end simulation

Why start-to-end simulation service?

- Estimation of experimental artifacts
- Experimental design and optimization
- Supporting material for beamtime proposals
- Assisting in decision making during an ongoing experiment

Photon Source

- *XFEL*
- *Synchrotron*
- *Optical Laser*

Photon propagation

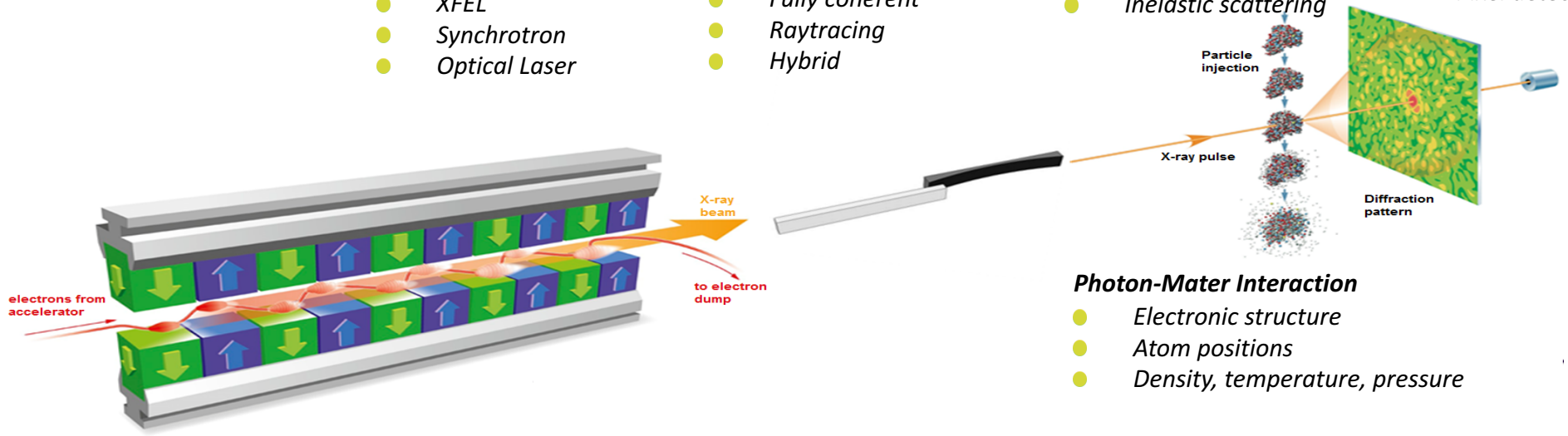
- *Fully coherent*
- *Raytracing*
- *Hybrid*

Signal generation

- *Spectroscopy*
- *Diffraction*
- *Inelastic scattering*

Detector response

- *Pixel detectors*



Photon-Mater Interaction

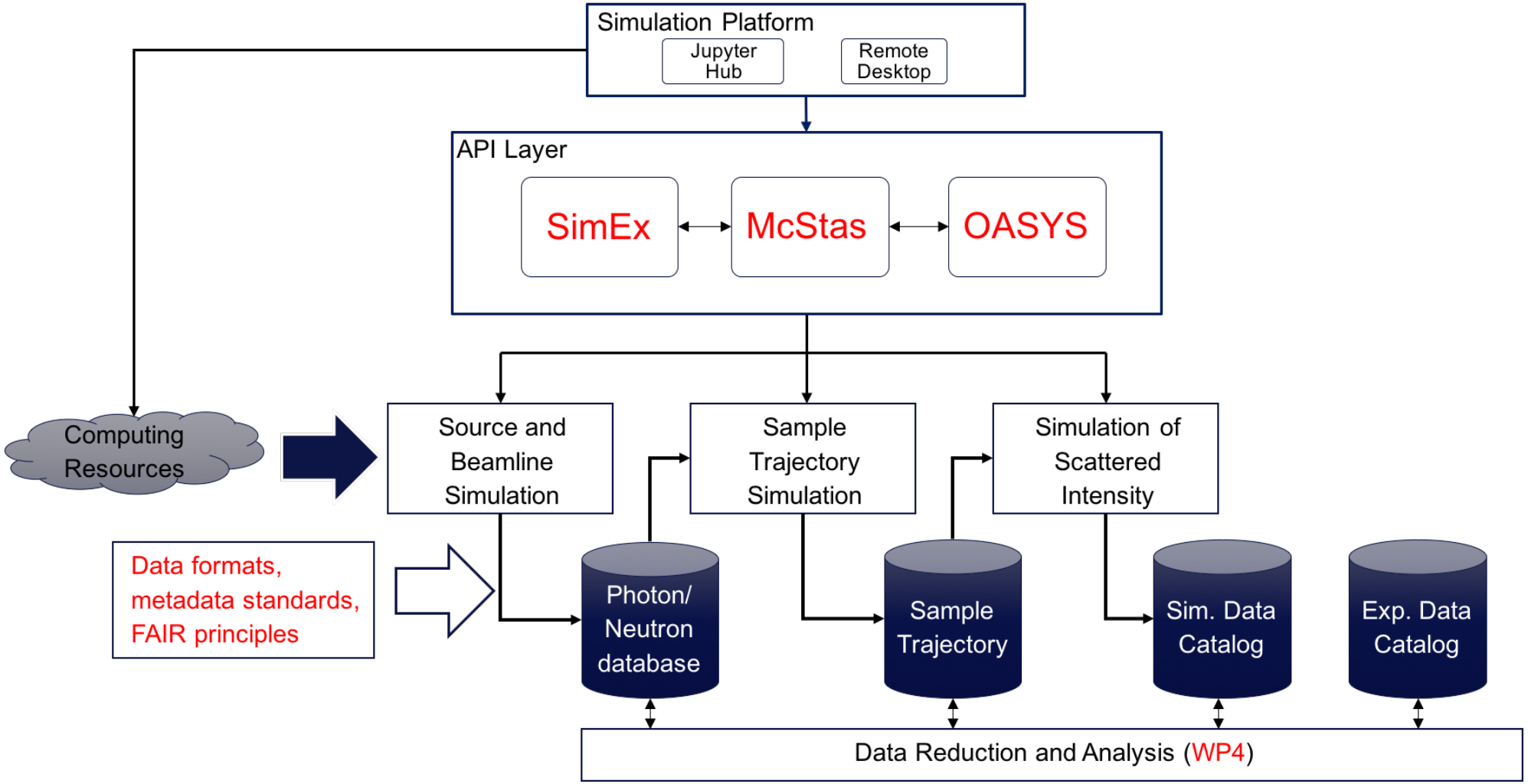
- *Electronic structure*
- *Atom positions*
- *Density, temperature, pressure*

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 823852

The Architecture

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 823852

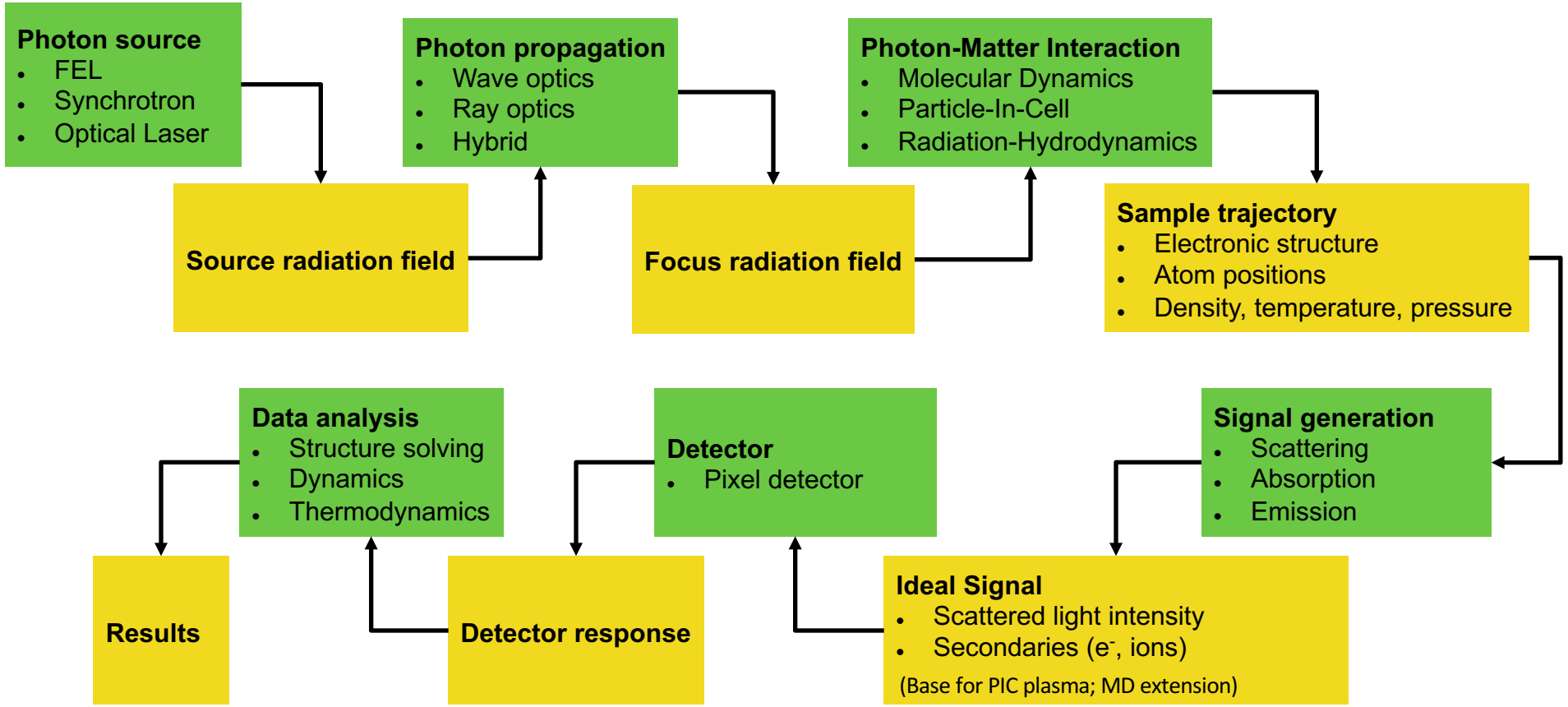
A harmonized API



This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 823852

SIMEX provides user interfaces and data formats for start-to-end photon experiment simulations

By J. C. E and CFG (EuXFEL)



Calculators: python APIs to advanced simulation codes

Data interfaces based on open metadata standards: openPMD

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 823852

Yoon, C. H. *et al. Scientific Reports* **6** (2016).
Fortmann-Grote, C. *et al. IUCrJ* **4**, 560–568 (2017).

SIMEX platform module list

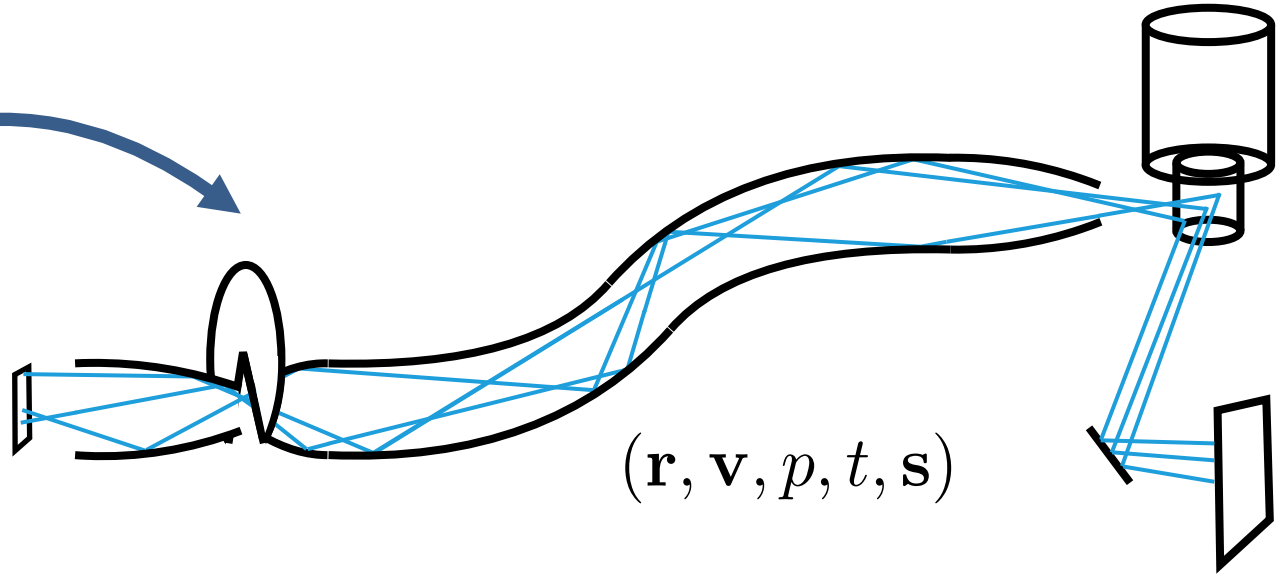
Module	Simulation type	Code	Collaborators
X-ray source	**FEL simulation**	**FAST-XPD**	**Manetti, Samoylova, Yurkov**
		GENESIS in OCELOT framework	S. Reiche / G. Geloni et al.
Propagation	**Coherent wavefront propagation**	**WPG/SRW**	**L. Samoylova, A. Buzmakov, O. Chubar**
X-ray Photon matter interaction	**Atoms, molecules, clusters**	**XMDYN & XATOM**	**Z. Jurek et al. (CFEL)**
		GROMACS	**I. Dawod et al. (U Uppsala)**
		HF/LDA	*H. Quiney et al. (U Melbourne)*
Optical photon matter interaction	1D Rad-Hydro	Esther	Colombier et al. (CEA)
	2D Rad-Hydro	*Multi2D*	*R. Ramiz et al.*
	3D PIC	*PiConGPU*	*M. Bussmann et al. (HZDR)*
Signal generation	**Molecule, cluster scattering**	**SingFEL**	**C. H. Yoon (LCLS)**
	Plasma SAXS	paraTAXIS	T. Kluge et al. (HZDR)
	Plasma Compton/Thomson	XRTS	G. Gregori, C. Fortmann-Grote
	Crystal diffraction	**CrystIFEL/pattern_sim**	**T. White et al. (CFEL)**
	Diffraction (large atomic system)	**GAPD**	**J. C. E, S. N. Luo et al.**
	EXAFS	FEFF8L	J. J. Rehr et al. (U Washington)
Detector simulation	2D Pixel detectors	X-CSIT, Karabo	T. Rüter et al. (EuXFEL)
Analysis/Reconstruction	**Pattern orientation**	**EMC**	**N. D. Loh (NUS, Singapore)**
	Phasing	**DM**	

Bold: developed actively; Normal: implemented, but not actively; Italic: planned for future interfacing

McStasScript

- McStas is a Monte Carlo neutron raytracing simulation tool
- Used at almost all neutron sources in the world for instrument design
- Written in C, user writes simulation in meta c language
- Selected to be included in PaNOSC harmonized simulation package
- Necessary to make a python API before this can happen, McStasScript

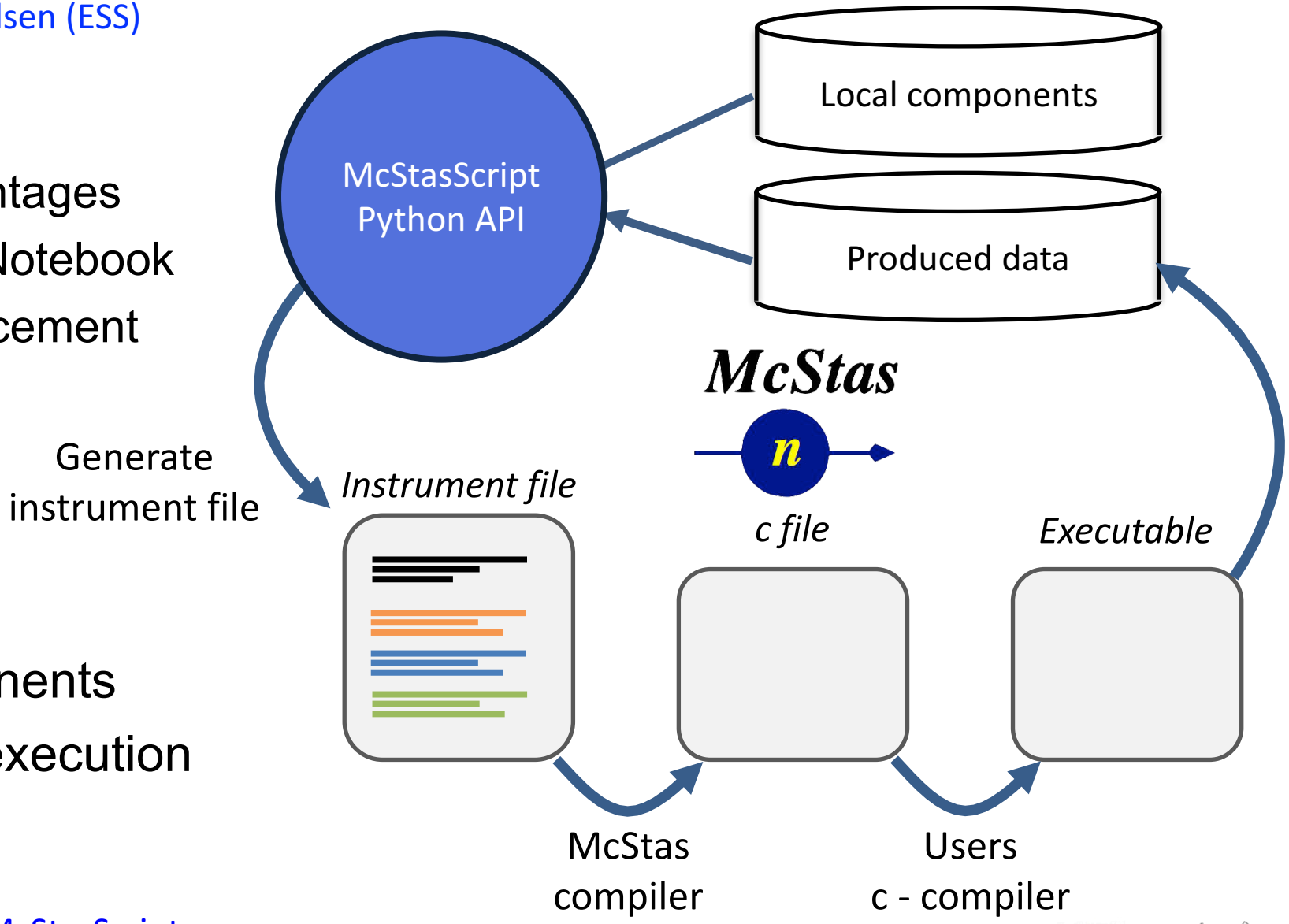
McStas simulation from source to detector including guide, chopper, sample environment and sample.



This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 823852

McStasScript By Mads Bertelsen (ESS)

- Python McStas API
  - Traditional Python advantages
  - Run McStas in Jupyter Notebook
  - More choice in code placement



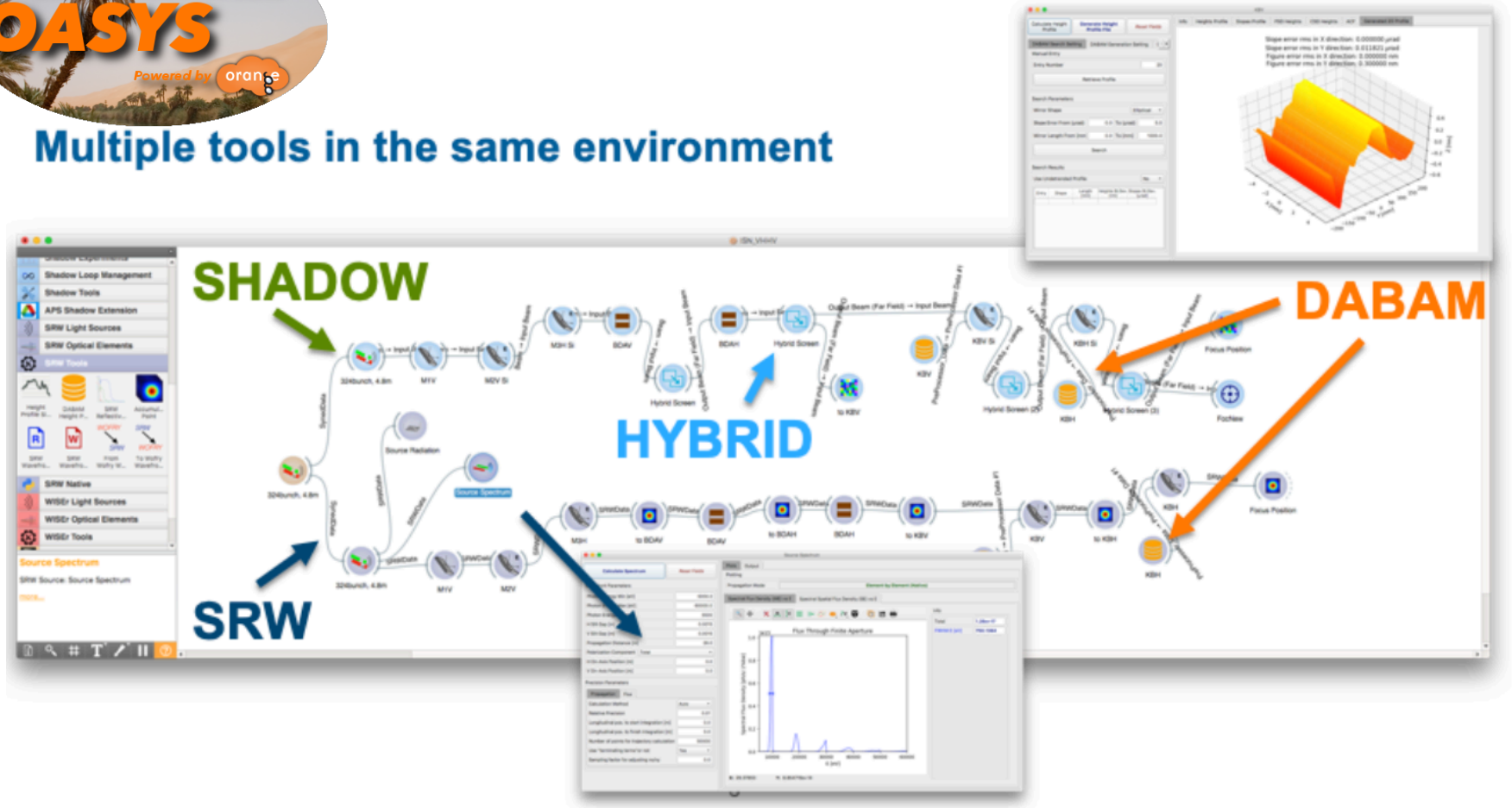
- Reads the local components
- Handles compilation / execution
- Handles import of data

https://github.com/PaNOSC-ViNYL/McStasScript

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 823852

OrAnge SYnchrotron Suite
OASYS
Powered by orange

Multiple tools in the same environment



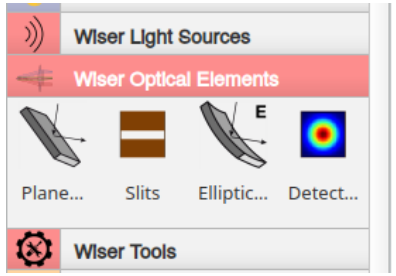
OASYS is a graphical environment for X-ray optics

Rebuffi, L. & Sanchez del Rio, M. OASYS (OrAnge SYnchrotron Suite): an open-source graphical environment for x-ray virtual experiments. in *Advances in Computational Methods for X-Ray Optics IV* (eds. Sawhney, K. & Chubar, O.) 28 (SPIE, 2017). doi:10.1117/12.2274263.

panosc

WISEr in OASYS By Aljosa Hafner (CERIC-ERIC)

- Port of WISE numerical integration code for wave propagation (Huygens-Fresnel principle): LibWiser – WofryWiser - OasysWiser
- Fast operation using numba kernel
- Details are in the other presentation also in this conference: Manfredda, M., Raimondi, L., Hafner, A., Mahne, N., Simoncig, A., Zangrando, M. WISEr x-ray wavefront simulation package: Advances and updates



master 1 branch 0 tags Go to file Add file Code

Aljosa Hafner Figure error changes, working 6aa455a 7 days ago 32 commits

orangecontrib	Figure error changes, working	7 days ago
tests	Split do_wise_calculation	2 months ago
LICENSE	Initial commit from Aljosa fork	4 months ago
MANIFEST.in	Rewiring of Wiser tools and updates to setup.py	last month
README.md	Update README.md	29 days ago
README.txt	Rewiring of Wiser tools and updates to setup.py	last month
setup.py	Figure error changes, working	7 days ago

README.md

Wiser for OASYS

OasysWiser is the GUI part of Wiser in Oasys.

Hierarchy:

- LibWiser - Python implementation of WISE (numerical integration using Huygens integral)
- WofryWiser - intermediate layer, taking care of compatibility with the rest of Oasys packages
- OasysWiser - GUI part, calling WofryWiser intermediate routines

Installation

About

The implementation of WISER into OASYS

wiser oasys

Readme

MIT License

Releases

No releases published
Create a new release

Packages

No packages published
Publish your first package

Contributors 2

aljosahafner aljosahafner

capitanevs capitanevs

Languages

Python 100.0%

Raimondi, L., and D. Spiga. Astronomy & Astrophysics 573 (Jan 2015): A22

This project has received funding from the European Union's Horizon 2020 research and inno

Interfacing data format standards for VINYL – openPMD standard

📖 http://www.openpmd.org

- Standardized hierarchical organization of (meta)data for fields and particles
- Open source text description

A example: the definition of the “geometry” attribute of mesh record in the openPMD base standard

The openPMD Standard

VERSION: **1.1.0** (Feburary 6th, 2018)

Conventions Throughout these Documents

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119.

All `keywords` in this standard are case-sensitive.

The naming *(floatX)* without further specification is used if the implementor can choose which kind of floating point precision shall be used (e.g. *(float16)*, *(float32)*, *(float64)*, *(float128)*, etc.). The naming *(uintX)* and *(intX)* without further specification is used if the implementor can choose which kind of (un)signed integer type shall be used (e.g. *(int32)*, *(uint64)*, etc.). The naming for the type *(string)* refers to fixed-length, plain ASCII encoded character arrays since they are the only ones that are likely to propagate through all file-format APIs and third-party programs that use them.

Sometimes brackets `<Name>` are used for keywords while the `<>` only indicate the keyword is required (but the `<>` itself shall not be written). Accordingly, optional keywords and options are indicated via square brackets `[Name]`.

The Versions of this Standard

Versions of the standard allow codes and implementors to easily differentiate between new updates that are incorporated.

The full version number is always used in format `<MAJOR>.<MINOR>.<REVISION>`.

Required Attributes for each `mesh record`

The following attributes must be stored additionally with each `mesh record` (which is a data set attribute for `scalar` or a group attribute for `vector` meshes):

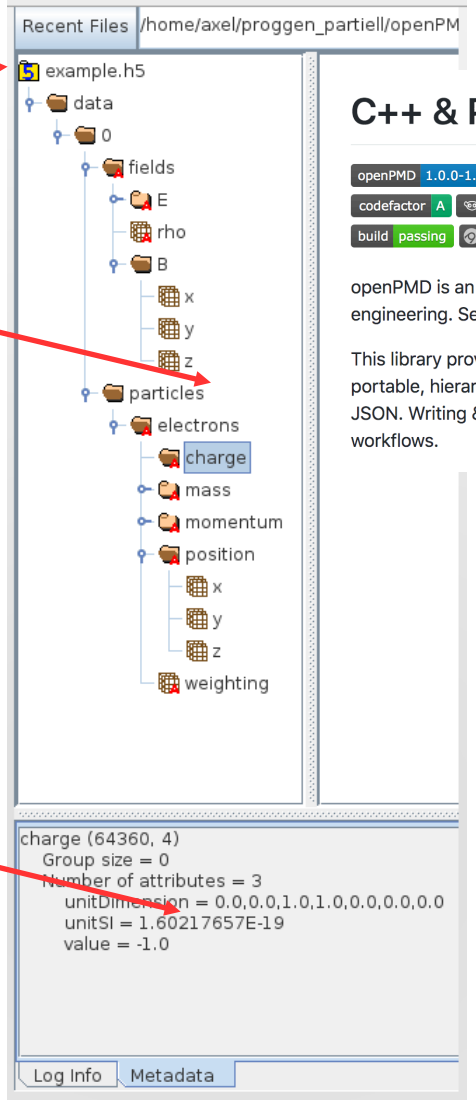
- `geometry`
  - type: *(string)*
  - description: geometry of the mesh of the mesh record, right-handed coordinate systems are imposed
  - allowed values:
    - `cartesian`: standard Cartesian mesh
    - `thetaMode`: regularly-spaced mesh in the r-z plane, with Fourier decomposition in the azimuthal direction (See doi:10.1016/j.jcp.2008.11.017) In this case, the mesh arrays are stored as a three-dimensional record where the last axis corresponds to the `z` direction, the second axis correspond to the `r` direction and where the first axis corresponds to the azimuthal mode. (This last axis has length `2m+1`, where `m` is the number of modes used. By convention, this first stores the real part of the mode `0`, then the real part of the mode `1`, then the imaginary part of the mode `1`, then the real part of the mode `2`, etc.)
    - reserved: `cylindrical`, `spherical`
    - `other`

[A. Huebl & M. Bussman, HZDR]

Interfacing data format standards for VINYL – openPMD standard

📖 http://www.openpmd.org

- Standardized hierarchical organization of (meta)data for fields and particles
- Independent of file format:
  - hdf5 serial
  - hdf5 parallel
  - adios1 and 2
  - Netcdf
- Flexible and open source
- Transparent support for physical units through standardized unit conversion scheme
- Application specific domain extensions:
  - PIC, wavefronts
  - Rad-Hydro
  - MD
  - QMD



C++ & Python API for Scientific I/O with openPMD

openPMD 1.0.0-1.1.0 | docs passing | API Doxygen | chat on gitter | platforms linux | osx | win | license LGPLv3 | DOI 10.14278/rodare.27
codefactor A | code quality: c/c++ A+ | code quality: python A+ | lgtm alerts 0 | coverage 84%
build passing | build passing | wheels passing | nightly packages failed | coverity passed

openPMD is an open meta-data schema that provides meaning and self-description for data sets in science and engineering. See the openPMD standard for details of this schema.

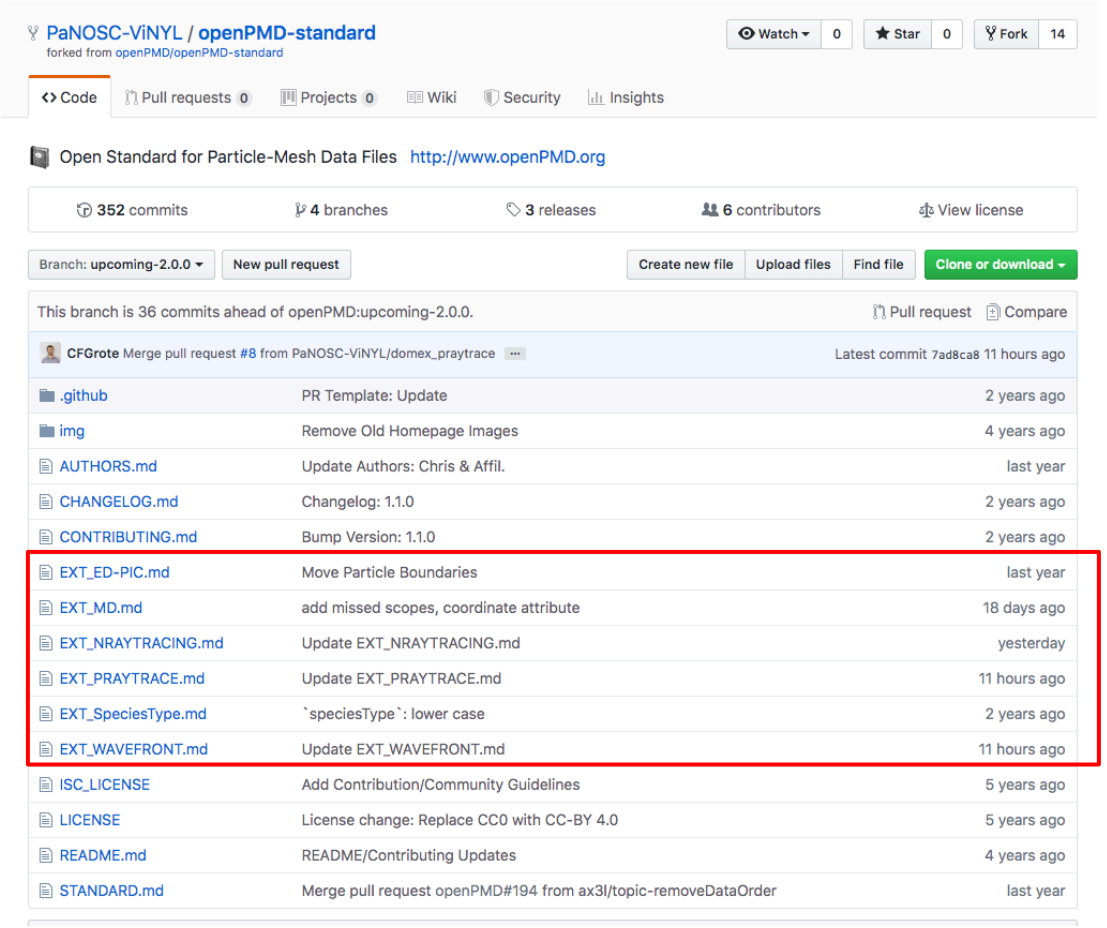
This library provides a reference API for openPMD data handling. Since openPMD is a schema (or markup) on top of portable, hierarchical file formats, this library implements various backends such as HDF5, ADIOS1, ADIOS2 and JSON. Writing & reading through those backends and their associated files is supported for serial and MPI-parallel workflows.

https://github.com/openPMD/openPMD-api

[A. Huebl & M. Bussman, HZDR]

Interfacing data format standards for VINYL

https://github.com/PaNOSC-ViNYL/openPMD-standard/tree/upcoming-2.0.0



- The members of VINYL has drafted OpenPMD domain extensions for:
  - Molecular Dynamics Simulation
  - Neutron raytracing
  - Photon raytracing
  - Coherent Wavefront Propagation

Interfacing data format standards for VINYL

https://github.com/PaNOSC-ViNYL/openPMD-standard/tree/upcoming-2.0.0

Domain-Specific Naming Conventions for Molecular Dynamics Simulation Codes

openPMD extension name: `MD`

VERSION: 0.0.2 (September 19th, 2019)

Introduction

This extension is specifically designed for the molecular dynamics simulation codes.

The current version of this extension is suitable to allow the output of arbitrary simulation codes to be post-processed and compared with common tools and frameworks. Future versions will define a common set of required records and further attributes.

The example data structure can be found HERE.

Root Group

Additional Attributes for the `Root` group

The following additional attributes are defined in this extension. The individual requirement is given in `scope`.

- `forceField`
  - type: 1-dimensional array of N (string) elements, where N is the number of force fields (interatomic potentials) implemented in the simulation.
  - scope: *optional*
  - description: the methods implemented in the simulation to describe the force fields (interatomic potentials). See Interatomic Potentials Repository.
  - example values:
    - `eam/alloy`
    - `lj/cut 3.0`
    - ...
- `forceFieldParameters`
  - type: 1-dimensional array of N (string) elements, where N is the number of force fields (interatomic potentials) implemented in the simulation.
  - scope: *optional*
  - description: the parameters specification for the `forceField` methods. See Interatomic Potentials Repository.
  - example values:
    - `pair_coeff * * 1 1`

- The members of VINYL has drafted OpenPMD domain extensions for:
  - Molecular Dynamics Simulation
  - Neutron raytracing
  - Photon raytracing
  - Coherent Wavefront Propagation

The Coherent Wavefront Propagation extension is under discussion with C. Mayes for better definitions

The Applications

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 823852

Gaussian beam

Setup beam parameters

```
beam_parameters = PhotonBeamParameters(
            photon_energy = 8.0e3*electronvolt,
            pulse_energy = 2.4e-6*joule,
            photon_energy_relative_bandwidth=1e-4,
            divergence=2.0e-6*radian,
            photon_energy_spectrum_type=None,
            beam_diameter_fwhm = None
            )
```

Save the HDF5 output

```
source = GaussianPhotonSource(parameters=beam_parameters,
                                        input_path="",
                                        output_path="gauss_source.h5")
source.backengine()
source.saveH5()
```

Read and plot the output

```
source_analysis = XFELPhotonAnalysis(input_path="GaussianSource.h5")
```

```
Start initialization.

Loading wavefront from GaussianSource.h5.
... done.

Getting intensities.
... done.
Data dimensions =  (400, 400, 20)

Masking NANs.
... done.
```

```
source_analysis.plotTotalPower()
source_analysis.plotTotalPower(spectrum=True)
source_analysis.plotIntensityMap()
```

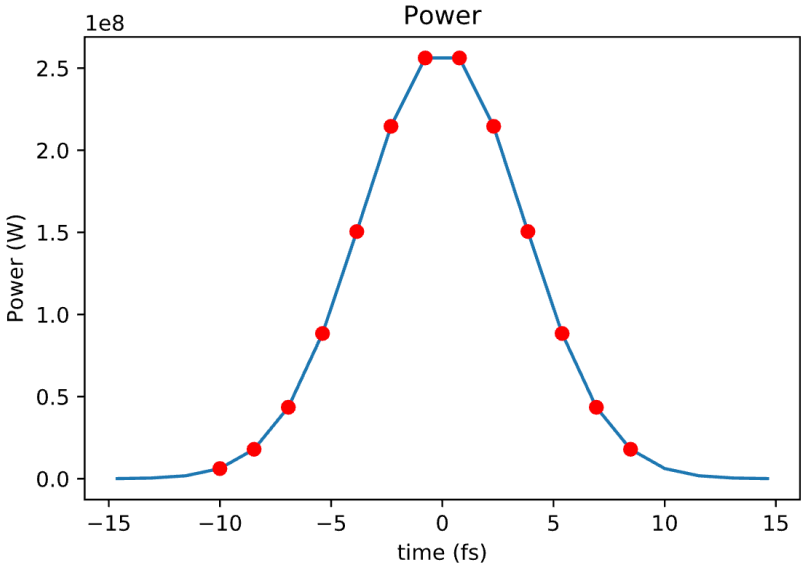
```
Plotting total power.
Pulse energy 2.4e-06 J
```

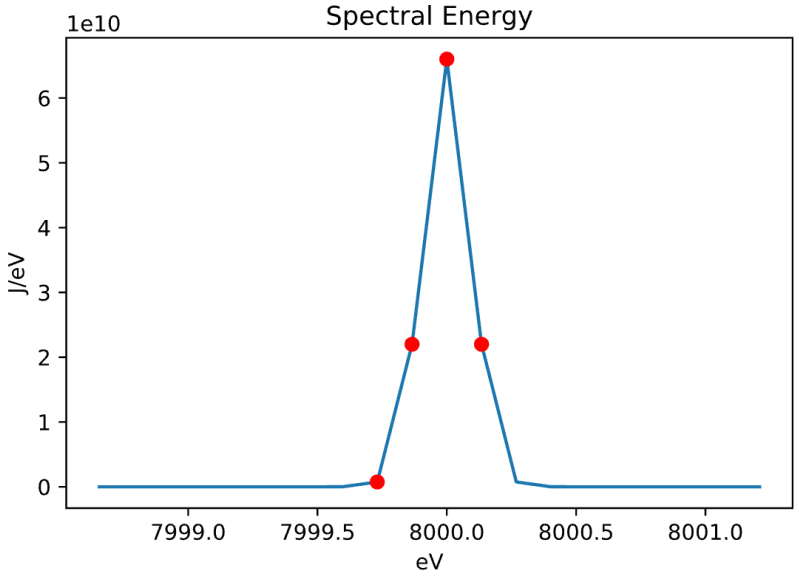
- A simple starting point for new users
- Prepare for wave propagation simulation

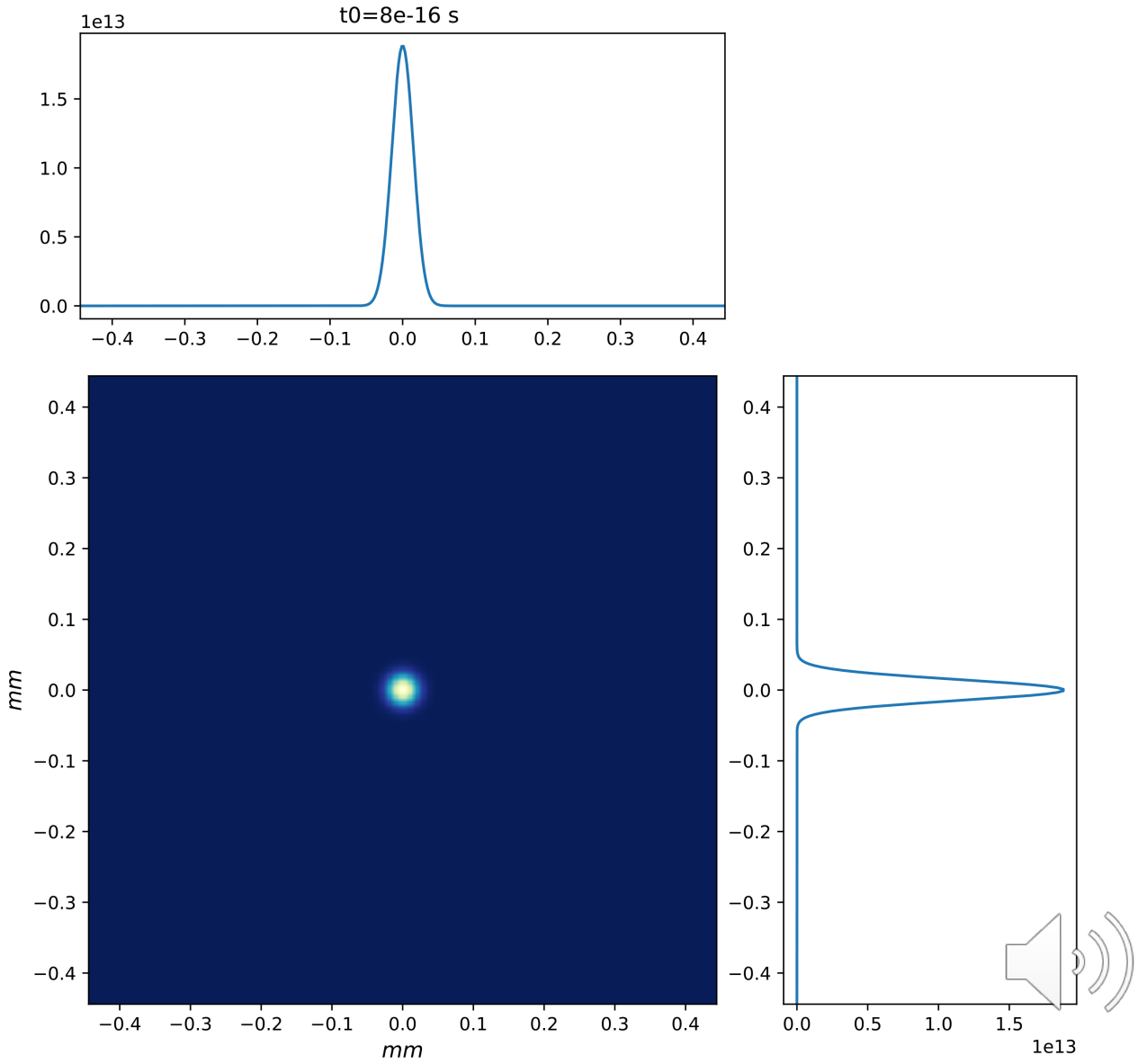
This project has received funding from the European Union's Horizon 2020 research and innovation programme

Gaussian beam

Simulation results







Gaussian beam

Propagate simulation with WPG

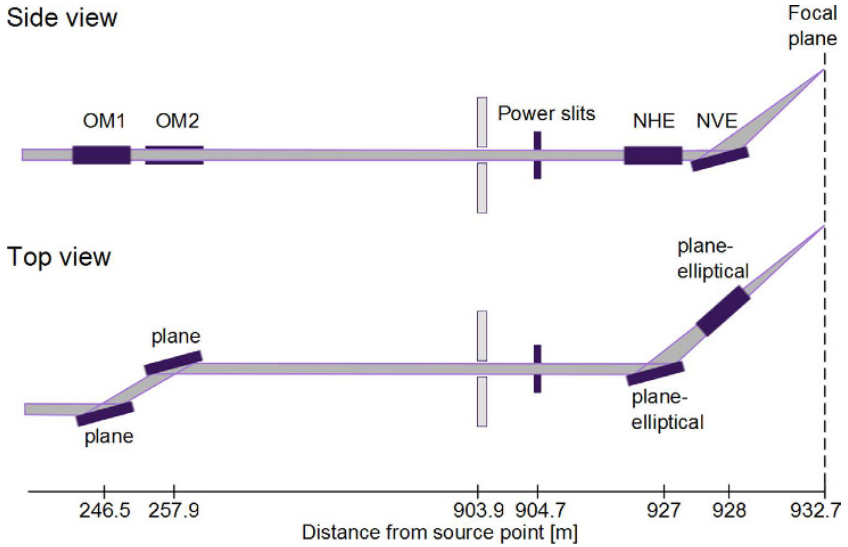
Import beamline for WPG

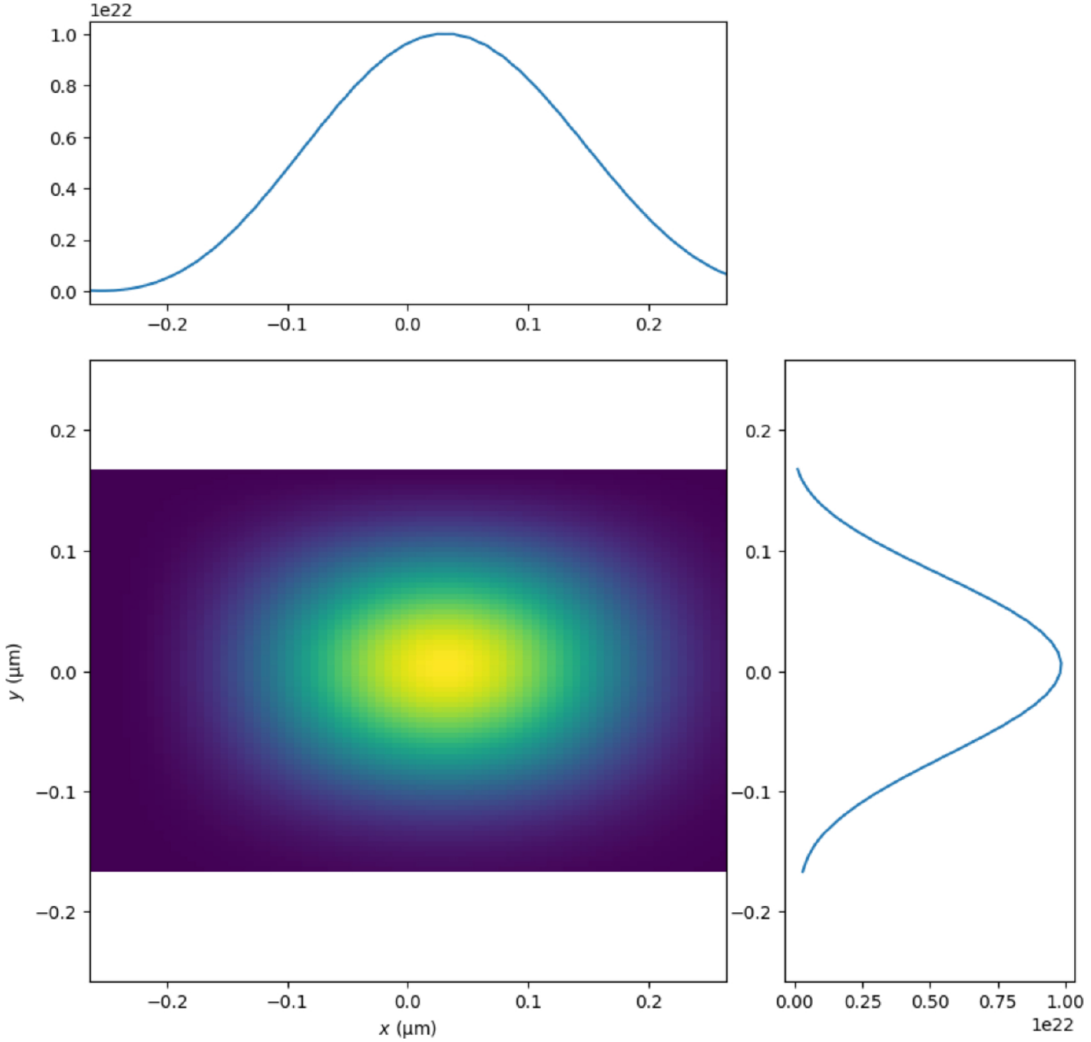
```
In [10]: from prop import exfel_spb_kb_beamline
```

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')
```



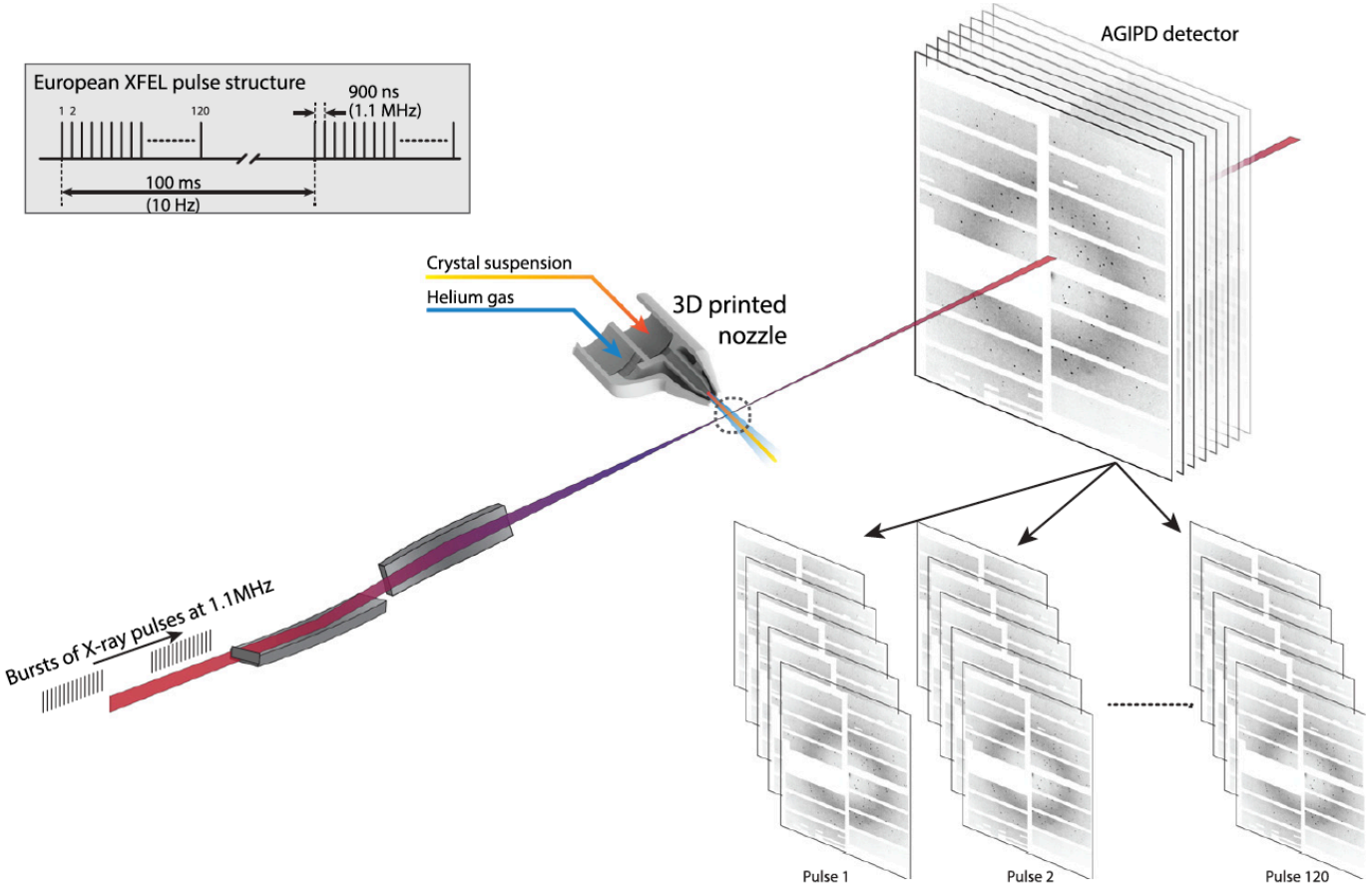


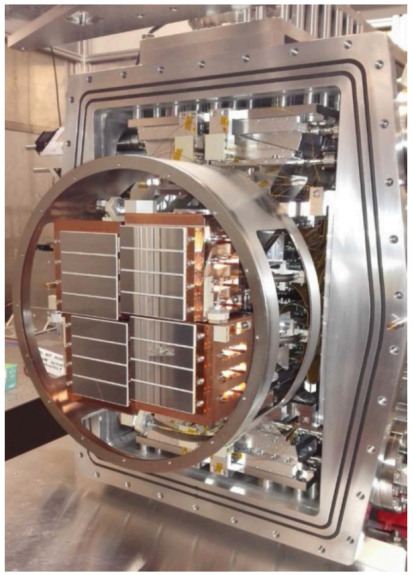
https://github.com/eucall-software/simex_notebooks/blob/master/start_to_end_demo.ipynb

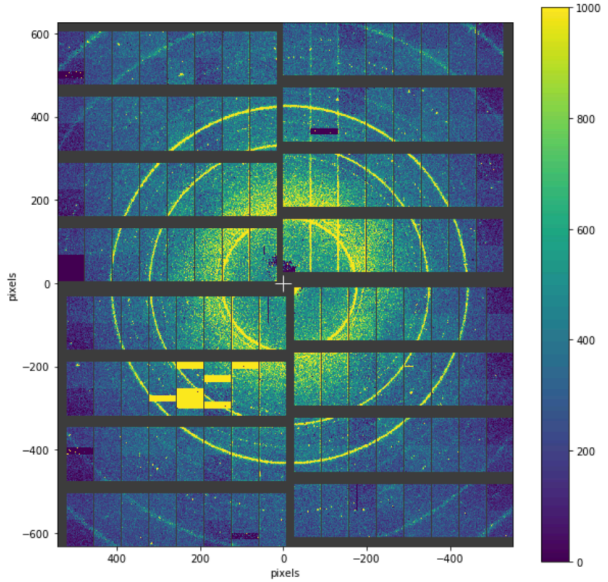
Yoon, C. H. *et al. Scientific Reports* **6**, (2016).

Serial crystallography simulation at European XFEL

- AGIPD 1M detector with multiple panel
- Millions frames of diffraction patterns of random oriented crystals collected and analyzed







Yefanov, O. *et al. Structural Dynamics* **6**, 064702 (2019).

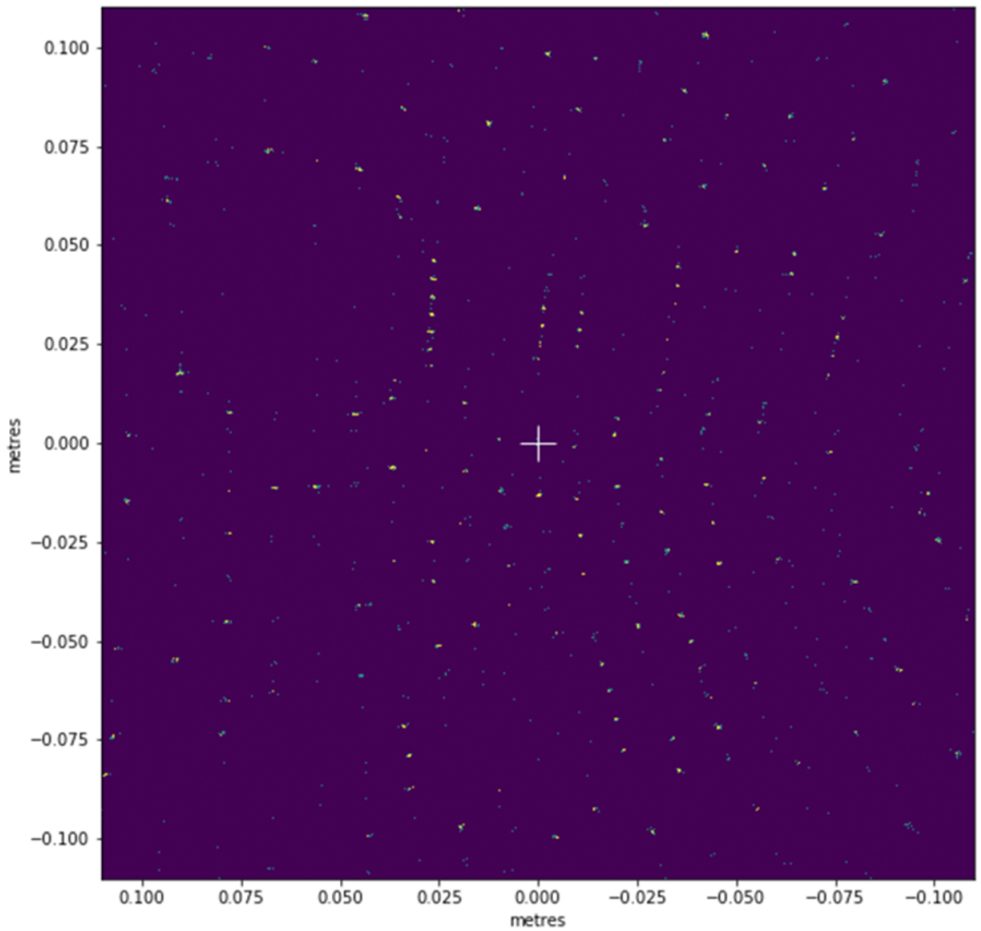
Serial crystallography simulation at European XFEL

Diffraction simulation with CrystFEL module

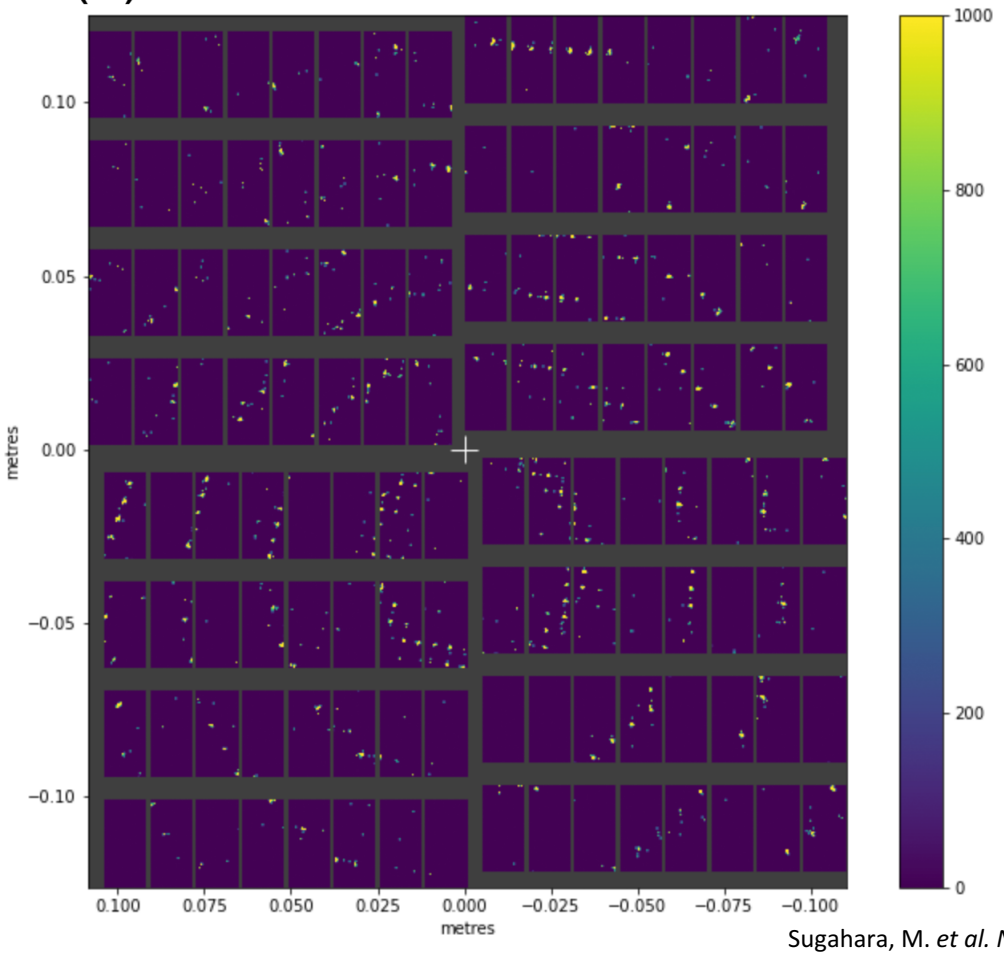
- Monolithic detector
- AGIPD detector with multiple panel
- Analyze the error due to the panel gaps

- Lysozyme sample (PDB: 3WUL)

(a)



(b)



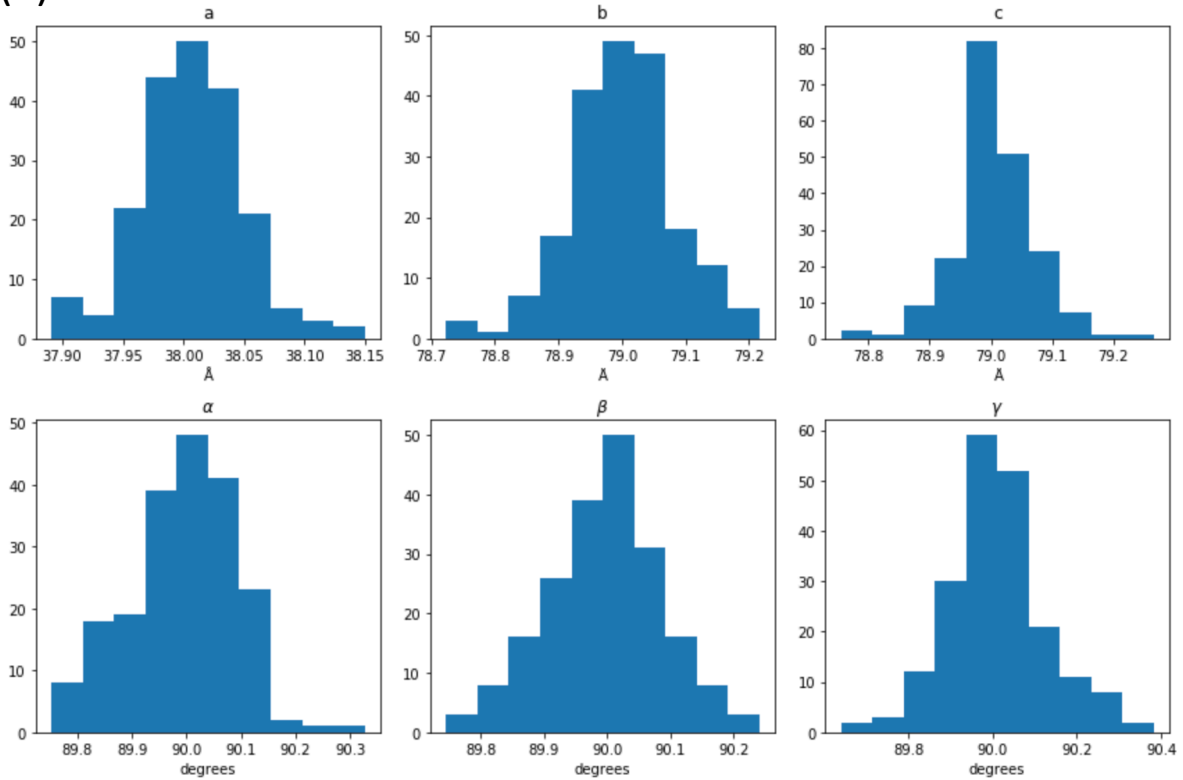
Sugahara, M. *et al. Nature methods* **12**, 61–63 (2015).

Serial crystallography simulation at European XFEL

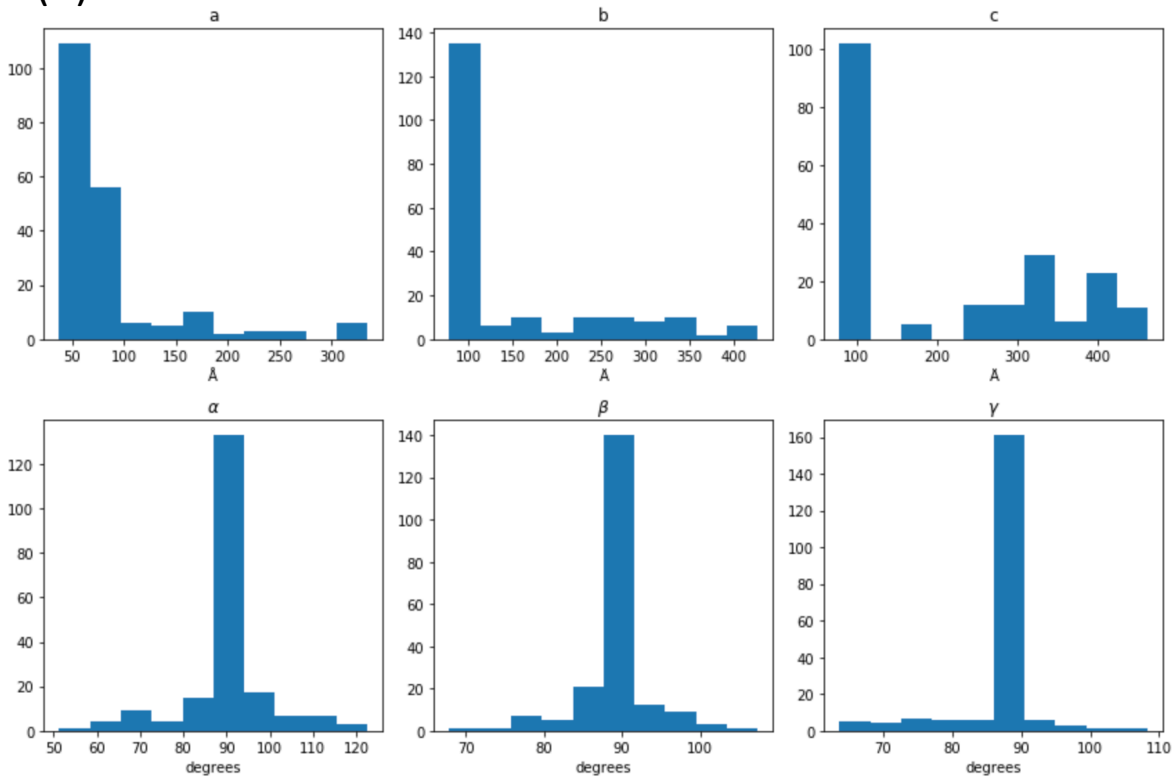
Analyze the simulated diffraction patterns

- Index the patterns and estimate the size and shape
- AGIPD detector with multiple panel
- A significant minority tail in the multiple panel case

(a)



(b)

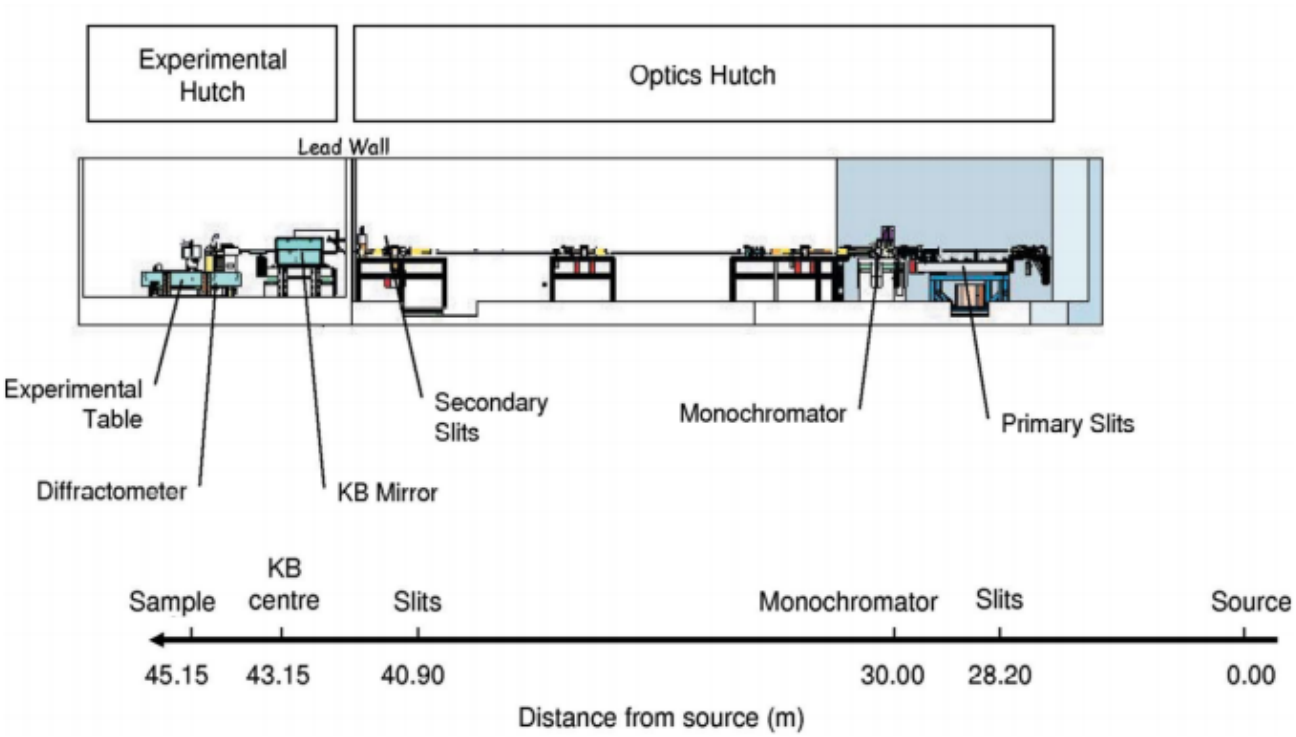


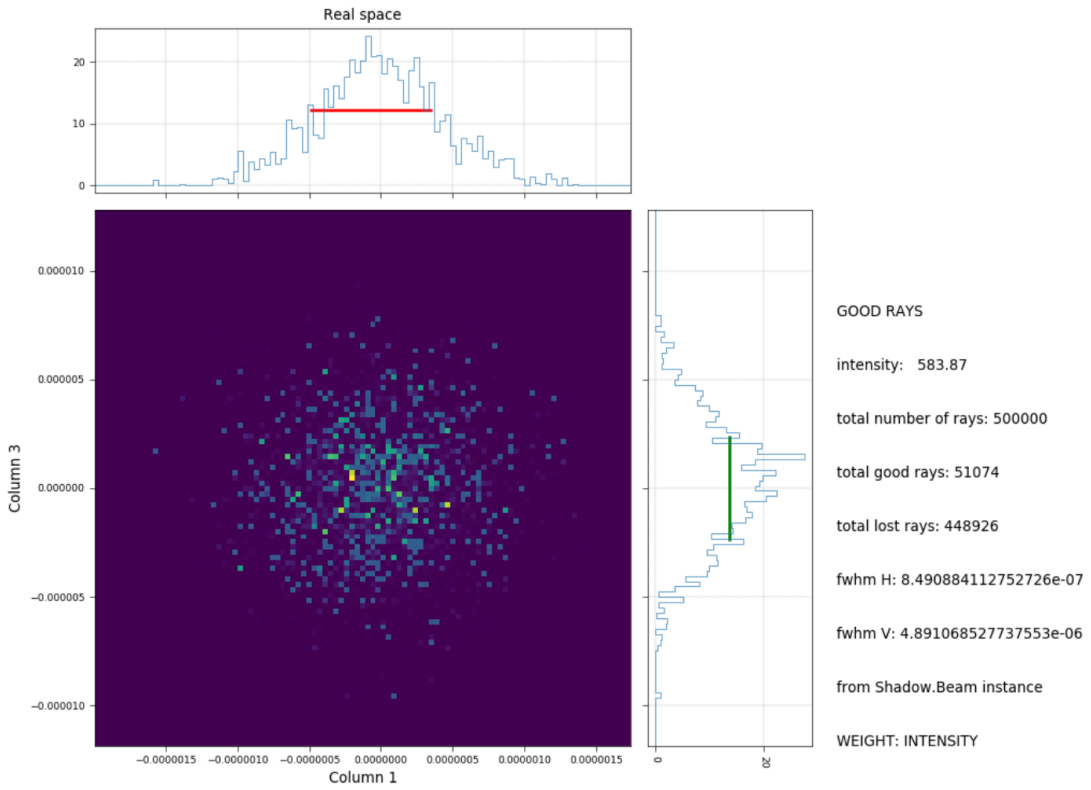
This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 823852

Polychromatic diffraction simulation using OASYS and SimEx

- ESRF ID23-2 beamline
- Simulation with relatively realistic X-ray spectrum
- Show the interoperability between OASYS and SimEx
- A good practice to bring a harmonized API

- 10^6 rays in the simulation





OASYS Shadow Raytracing

Flot, D. et al. Journal of synchrotron radiation 17, 107–118 (2010).

Polychromatic diffraction simulation using OASYS and SimEx

Interface: Jupyter notebook

Setup detector parameters

```
detector_panel = DetectorPanel(
            ranges={
                'fast_scan_min': 0,
                'fast_scan_max': 200,
                'slow_scan_min': 0,
                'slow_scan_max': 200
            },
            pixel_size=2200e-6 * meter,
            photon_response=1.0,
            distance_from_interaction_plane=0.25 * meter,
            corners={
                'x': -100,
                'y': -100
            },
        )
detector_geometry = DetectorGeometry(panels=[detector_panel])
```

Read raytracing output generated by OASYS

Polychromatic Beam setup

```
beam = './raytracing_out.h5'
```

Diffractor setup

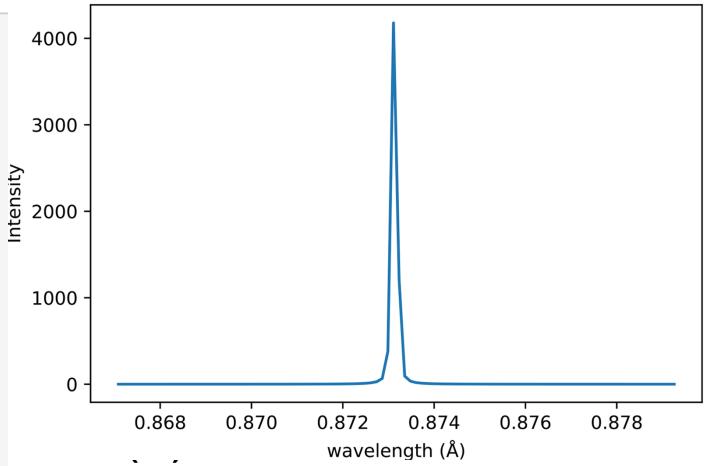
Setup diffractor parameters

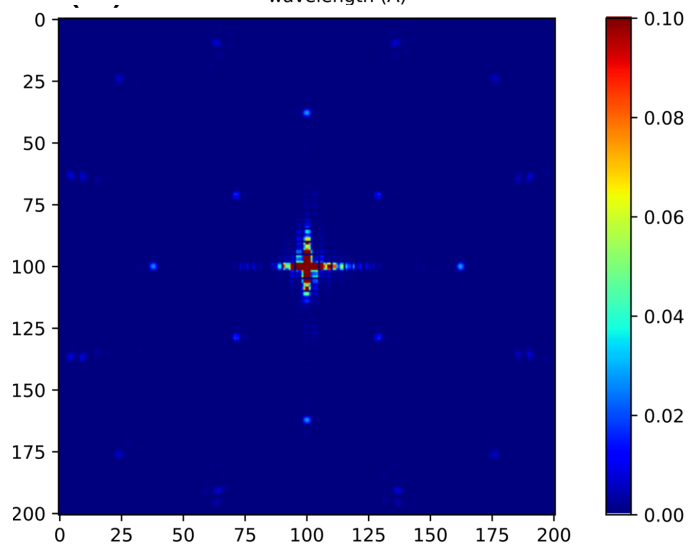
```
outfile = 'diffr_poly_1.txt'

parameters = GAPDPhotonDiffractorParameters(
            detector_geometry=detector_geometry,
            beam_parameters=beam,
            number_of_spectrum_bins = 100)

diffractor = GAPDPhotonDiffractor(parameters=parameters,
                                  input_path='single-cu.xyz',
                                  output_path=outfile)
```

Passing beam parameters as a tratracing_out file.

Run the diffraction backengine

```
diffractor.backengine()
```





SimEx

Summary

- **VINYL will provide a harmonized API for Photon and Neutron facilities to conduct start-to-end simulation through:**
  - Backends of SimEx, OASYS and McSats
  - openPMD format standard
  - Interoperability
- **VINYL will help:**
  - estimate experiment artifacts and their effects on analysis
  - design and optimize experiment setups
  - generate supporting material for beamline proposals

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 823852

photon and neutron
open science cloud

Thank you

Juncheng.e@xfel.eu

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 823852