

PAUL SCHERRER INSTITUT

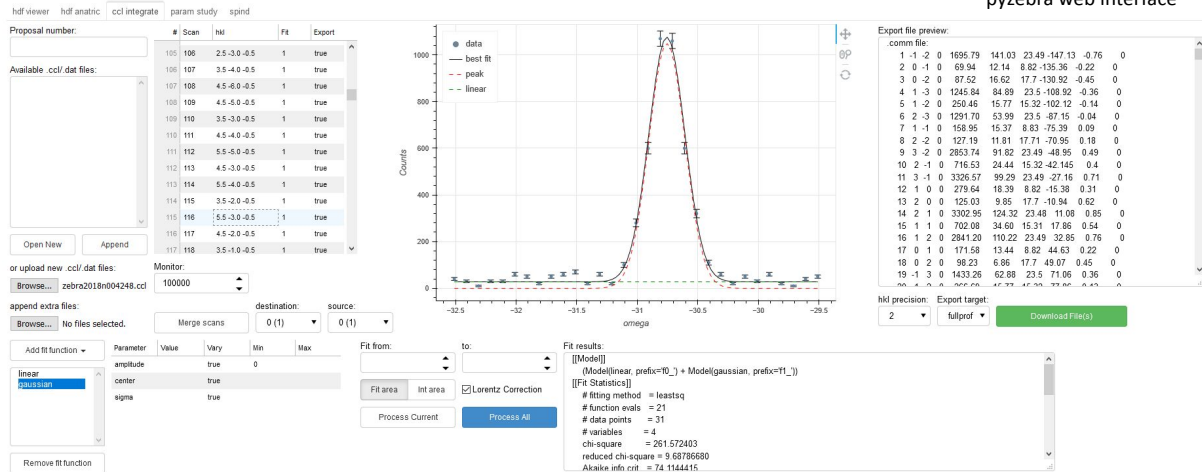


Ivan Usov :: Paul Scherrer Institute

pyzebra

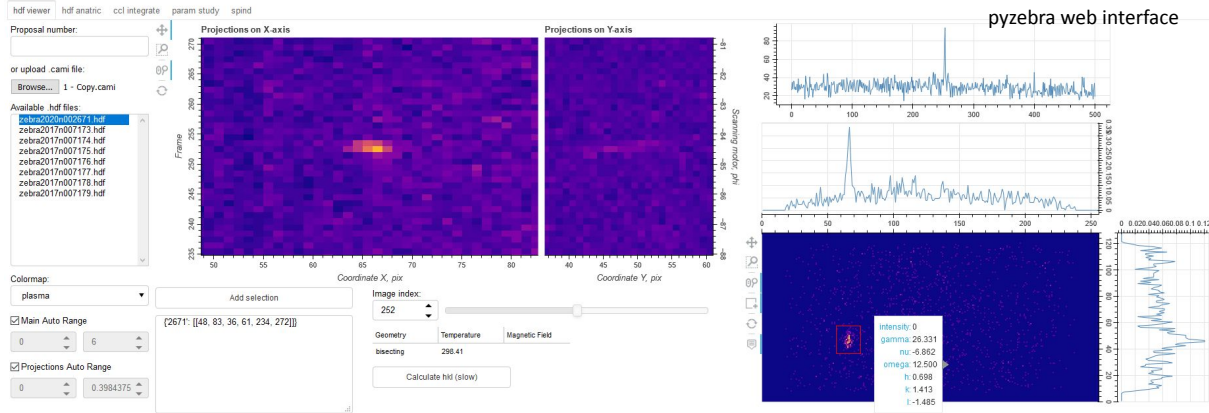
17 Aug 2022

Project description



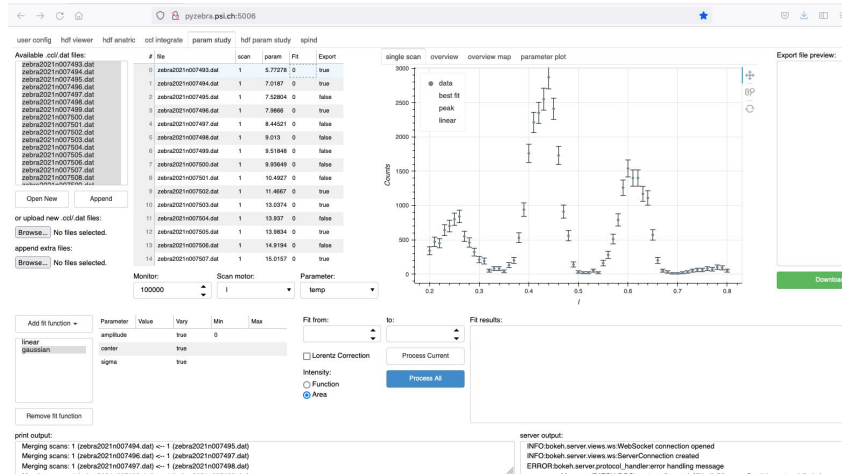
zebra instrument

- pyzebra is a python data analysis library and a web app for the zebra instrument at SINQ
- It includes a set of tools like data file viewers, standard statistics and analysis methods, as well as wrappers for 3rd-party applications (e.g. anatic, spind)



crystal sample
studied at zebra
beamline

- The design goal is to make it convenient for users of all backgrounds to process data and pipe results from raw to a desired output format, which before included multiple processing steps in different programs, as well as to do a more elaborate analysis for staff members during beamtimes
- The web service can be used from any OS (Windows, Mac, Linux) and any major browser (Firefox, Chrome, Safari), and doesn't require users to install any specific dependencies



- The pyzebra web interface is in production during zebra beamtimes since October 2020 and we continuously improve the service quality based on users' feedback
- We are also committed to improve the underlying python library to enhance functionality in jupyter notebooks for non-standard data analysis pipelines
- The project is in a phase where most of the expected functionality is ready, but going forward, it still requires occasional edge-case handling and user support

Starting with a sketch

STEPS

user config hdf viewer hdf anatic **cci prepare** cci integrate cci compare param study hdf param study spind

1 **Geometry:** bisecting or open CFL: browse
 normal beam

2 **Wavelength:** Max. Angular limits: stt/ga Min.

Open CIF: browse or open GEOM: browse omega
chi/nu
phi

Crystal structure: space group
cell

3 **UB matrix:**

4 **Ranges: HKL** **sin(theta)/lambda**

5 **Magnetic structure (optional):** lattice **GO**
k vector **more**

6 **Sorting:** 1st Δ 2nd Δ 3rd **GO**

7 **Created lists:**
list.hkl
list.mhkl

Preview selected list:

l	h	k	F2	Gamma	Omega	Nu
0	-2	3	0.00000	7.814	-74.010	13.788
0	2	-3	0.00000	7.814	105.990	-13.788
...						

Download file **Plot selected list**

Measured data: browse .comm/.incomm files **Plot selected file**

interactive plot
(3D view rotation with click+mouse)

8

HKL normal delta in-plane X Y

Distinguish options: files (symbols)
 intensities (size)
 k vectors nucl/magn (colors)
 scan direction resolution ellipsoid **Clear plot**

LEGEND:

- selection button (mutually exclusive)
- selection button (mutually exclusive if one the same line)
- numerical field for manual entry and/or suggested default
- dropdown menu for numerical input
- more** adds a numerical input field

GO

UB matrix:

plot reciprocal space

} runs Fortran program or python script

Add list to reciprocal space plot re-runs plotting script using new and previous lists as inputs

Clear plot clears all current entry lists of plotting script

Starting with a sketch and a description...

STEPS

- 1 **Geometry:** I O
- 2 **Wavelength:**
- 3 **Open CIF:**
- 4 **Crystal structu**
- 5 **UB matrix:**
- 6 **Ranges:** HKL
- 7 **Magnetic struc**
- 8 **Sorting:** 1st
- 9 **Created lists**
list.hkl
list.mhkl
- 10 **Preview select**
I h k l
0 -2 3
0 2 -3
...
Download file

1 .geom input for Sxtal_Refgen program (Fortran)

- click on bisecting or normal beam opens the corresponding `zebraBI.geom` or `zebraNB.geom` files (input for `Sxtal_Refgen`)
- angular limits are loaded from `.geom` files (default values) but can be edited in the corresponding numerical fields → `ANG_LIMITS` in `Sxtal_Refgen`
- alternatively, one may directly browse and open a `.geom` file, in which case `ANG_LIMITS` is read from there

2 .cif input for Sxtal_Refgen program (Fortran)

- wavelength needs either to be specified from standard values in the dropdown menu (0.788, 1.178, 1.383, 2.305) or manually entered → `WAVE` in `Sxtal_Refgen`
- either browse CIF and extract information from there (space group, cell and atomic positions) or user inputs manually the space group and cell → `CELL` and `SPGRP` in `Sxtal_Refgen`
- if CIF, relevant keywords: `_space_group_name_H-M_alt` → `SPGRP` in `Sxtal_Refgen`

```

_cell_length_a _cell_length_a _cell_length_a _cell_length_alpha _cell_length_beta _cell_length_gamma → all in one line, called CELL in Sxtal_Refgen
_atom_site_label
_atom_site_type_symbol
_atom_site_fract_x
_atom_site_fract_y
_atom_site_fract_z → all in one line, called ATOM in Sxtal_Refgen (one line per atom in the cell...), example: ATOM Pt1 PT 0.0 0.0 0.0 0.00000 1.00000
_atom_site_U_iso_or_equiv
_atom_site_occupancy
    
```

maybe CIFs are inconvenient to extract atomic positions? is it easier to first convert to binary?
<https://github.com/dsehnal/BinaryCIF>

- alternatively, one may directly browse and open a `.cif` file, in which case all information (`WAVE`, `CELL`, `SPGRP`, `CELL`, `ATOM(s)`) is read from there

3 .cif input for Sxtal_Refgen program (Fortran)

- `Sxtal_Refgen` requires a UB matrix in order to run → `UBMAT` in `Sxtal_Refgen`
- the matrix may be specified by the user by pasting a line of nine comma-separated float numbers in the numerical field: `ub11, ub12, ub13, ub21, ub22, ub23, ub31, ub32, ub33`
- however, sometimes we want `Sxtal_Refgen` to generate an initial UB matrix, which can be done by running the program using the correct `CELL` and `SPGRP` (and a default and therefore wrong `UBMAT`)
- this dummy run of `Sxtal_Refgen` should happen when clicking on the `UB matrix` button (the server should then read the matrix calculated from the cell parameters from the `.sfa` output file of `Sxtal_Refgen` – see slide 4, and make it appear in the UB matrix numerical field)
- (remark: the UB format in one line is necessary because it is the format for the instrument control software from which we copy/paste matrices, however, in `Sxtal_Refgen` the same matrix is in space-separated 3x3 format)

4 .cif input for Sxtal_Refgen program (Fortran)

- the user may want to edit the limits in HKL and in `sin(theta)/lambda` → `HLIM` and `SRANG` in `Sxtal_Refgen`
- the corresponding numerical fields should show the default values: `HLIM (-25 25 -25 25 -25 25)` and `SRANG (0.0 0.7)`
- the program should then run with all the above inputs when clicking on the first `GO` button

5 .cif input for Sxtal_Refgen program (Fortran)

- `Sxtal_Refgen` may run with or without inputs from the user regarding a magnetic structure, these are the magnetic lattice and propagation vector → `lattICE` and `kvect` in `Sxtal_Refgen`
- several propagation vectors may be added, by adding corresponding numerical fields using the `more` button
- if no input from the user on these optional fields, then the program should run on the default values from the `.cif` file, which don't need to be displayed in the numerical field

...To a tab with the new functionality

user config hdf viewer hdf anatic **ccl prepare** ccl integrate ccl compare param study hdf param study spind

Open CFL: No file selected. Open CIF: No file selected. Open GEOM: No file selected. Measured data: No files selected.

Geometry:
 bisecting
 normal beam

Wavelength:
 preset value

Angular min/max limits:
 stt/gamma omega chi/nu phi

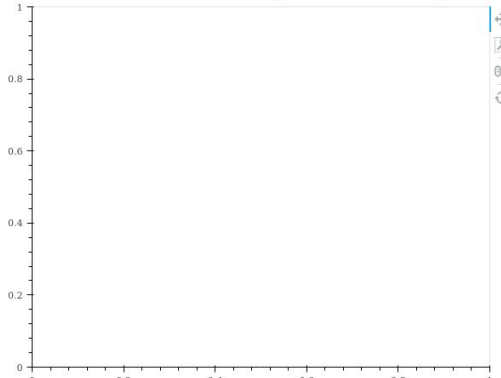
Crystal structure:
 space group cell

Ranges:
 HKL $\sin(\theta)/\lambda$

Magnetic structure:
 lattice k vector

1st Δ 2nd Δ 3rd Δ

Created lists:
 Preview selected list:



HKL:
 normal cut delta in-plane X in-plane Y

k vectors:

Distinguish options:
 files (symbols)
 intensities (size)
 k vectors nucl/magn (colors)
 scan direction
 resolution ellipsoid

Resolution mult:


print output:

server output:
 INFO:bokeh.server.views.ws:WebSocket connection opened
 INFO:bokeh.server.views.ws:ServerConnection created

Deployment and references

We run 2 instances (systemd units) on a VM at PSI:

- <http://pyzebra.psi.ch/> - prod, it receives updates only outside of beamtimes and is used as a stable implementation (tracks latest git version tag)
- <http://pyzebra.psi.ch:5006/> - test, for receiving feedback from users on new features and hotfixes, but can be restarted any moment (tracks latest git commit)

The web app implementation is based on Bokeh (<https://bokeh.org/>)  **bokeh**
- a Python library for creating interactive visualizations for modern web browsers

Github project link - <https://github.com/paulscherrerinstitute/pyzebra>

In collaboration with

- Holzer Jakob
- Larsen Camilla Buhl
- Sibille Romain Franck
- Zaharko Oksana

