
WIR SCHAFFEN WISSEN – HEUTE FÜR MORGEN

Anders Kaestner :: Imaging scientist :: Paul Scherrer Institut

Open source tools for analyzing neutron imaging data

ILL-ESS topical meeting, ~~Grenoble~~ Online, 14-15 October, 2020

About me and my roles

Instrument scientist

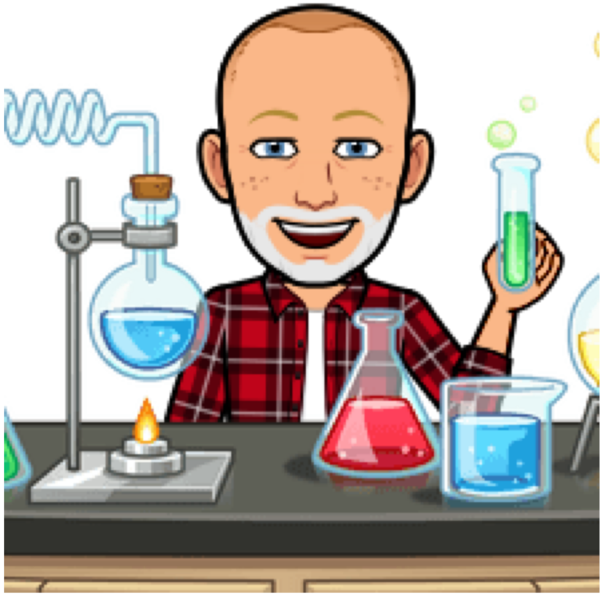


Image processing & software developer

Introduction

What happens after an imaging experiment?

Gigabytes. . .

Image acquisition is not the end of the experiment

Collaboration with EUROPEAN SPALLATION SOURCE

What do we want to know?

Find hidden objects

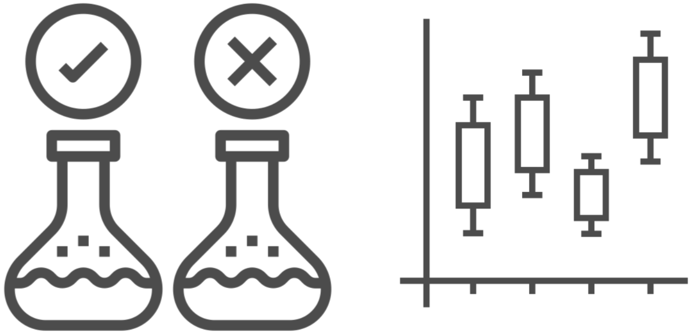
Dimensions

Counting objects

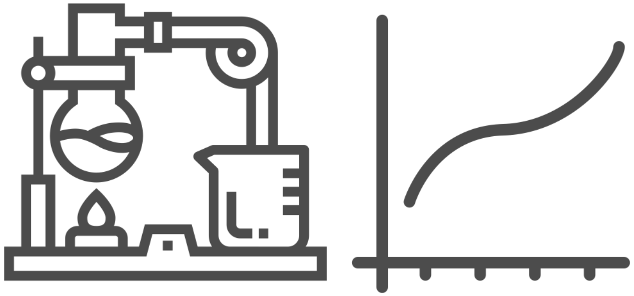
Mixing ratios

Track changes

What do we *really* want to know?



Compare treatments

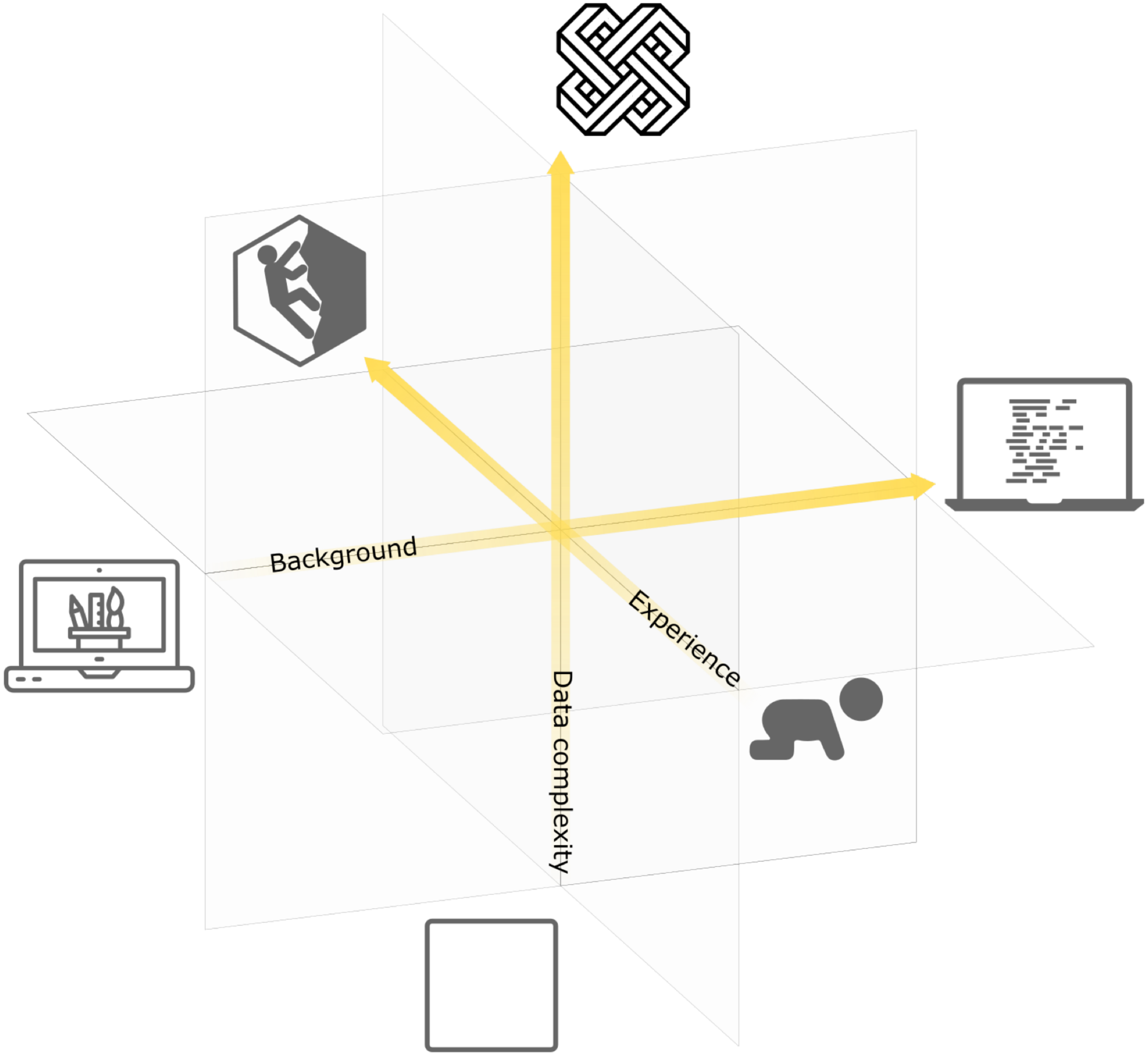


Understand and model processes

Inspect components

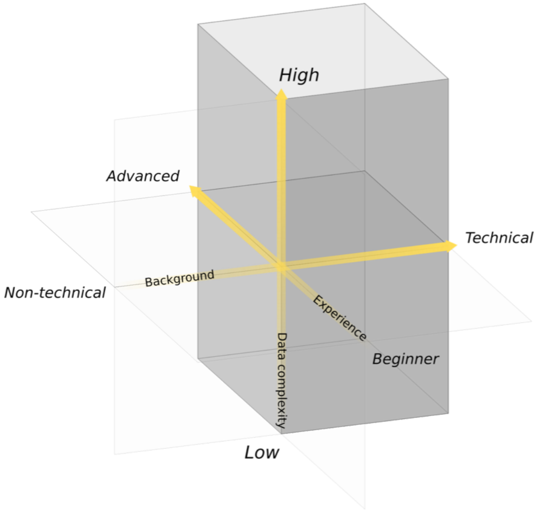
Data evaluation fitness of users

Users of neutron imaging have very different background and experience

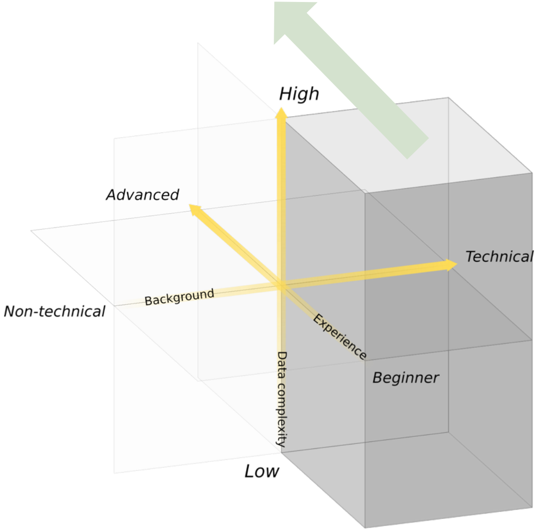


Data evaluation fitness of users - Characterization

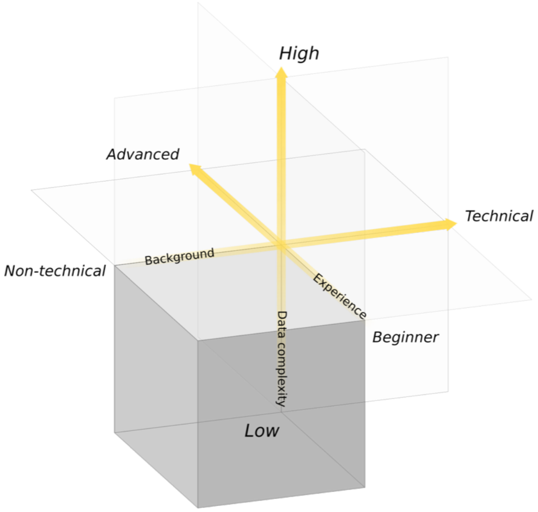
Beamline scientist



Early stage PhD student



Cultural heritage users

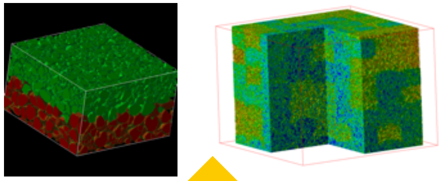


Serving people with different background and needs is a great challenge

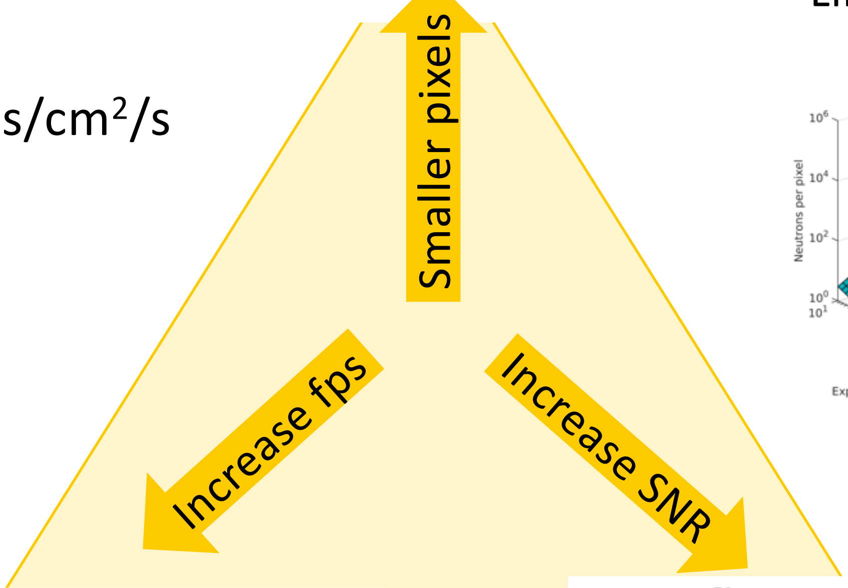
How to analyze the data

Challenge of neutron imaging and the analysis

Feature size vs resolution



About 10^7 neutrons/cm^2/s

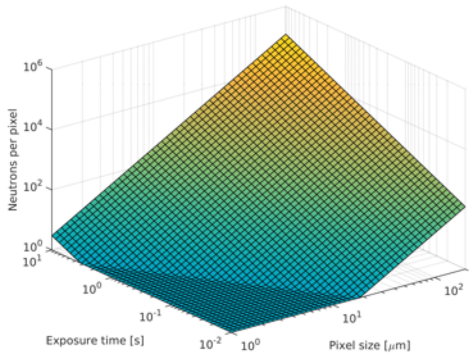


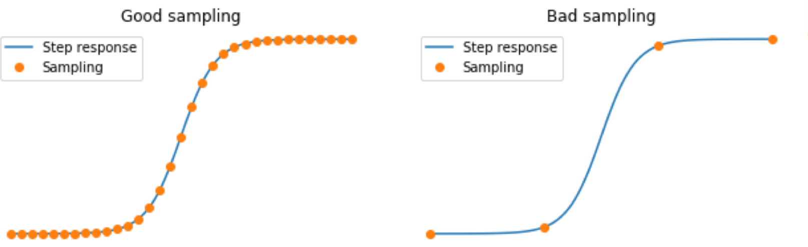
Smaller pixels

Increase fps

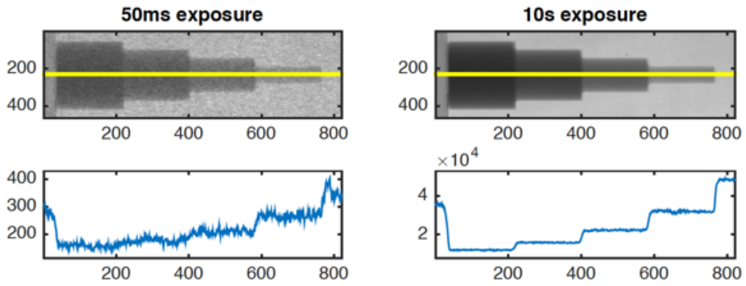
Increase SNR

Limitation: Neutron flux





Process speed vs. Frame rate

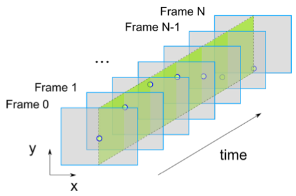


Contrast vs noise

Data processing

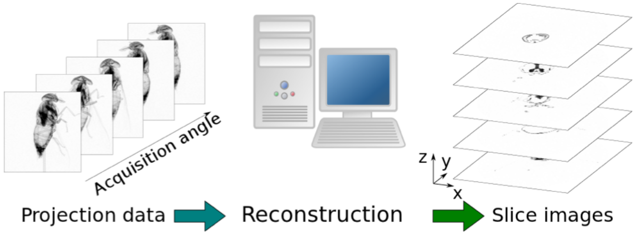
Data processing can be a very labor- and computationally intense task

Basic processing



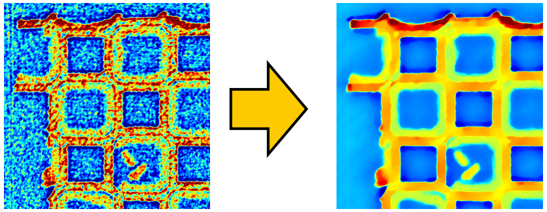
Minutes, hours

CT reconstruction



Minutes, hours

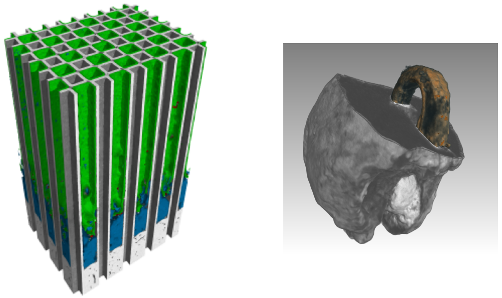
Image processing



Hours

Visualization

Analysis

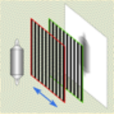


Hours, days

Synergies in the analysis workflow

Most experiments do (some of these):

- Basic processing
  - Normalization
  - Artifact removal
  - Scattering correction
  - Denoising
- Transformations
  - CT reconstruction
  - nGI reduction
  - Bragg-edge fitting
- Geometric transformations
  - Stitching
  - Registration



Experiment specific analysis

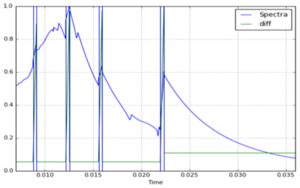
- Segmentation/Classification
- Modelling
- Feature analysis
  - Dimensions
  - Counting
- Etc

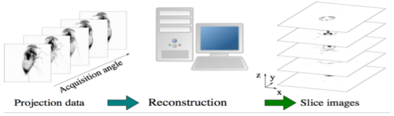
→ Template workflows

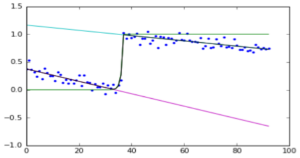
New challenges: 4D imaging

- Artefact reduction
- Re-binning (time/space)

- Denoising
- Segmentation







ToF CT

- 1000x1000 pixels
- 1000 projections
- 2000 ToF bins

4Tb

CT

- 1000^3 voxels
- 2000* ToF bins

4Tb

Bragg edge fitting

- 1000^3 voxels
- 2000* ToF bins
- 7 parameters/edge

N x 7 x 4Tb

- CLI tools
- Python
- Mantid
- KipTool

- MuhRec
- (Octopus)
- Python/Matlab
  - ASTRA
  - TomoPy

- RITS
- iBeatles
- ToFImaging (under development)

* 2000 volumes is a worst case scenario.

* 2000 volumes is a worst case scenario.

Scalability – Computing infrastructure needed

Laptop	Workstation	Cluster
		
Pre-evaluation Training sessions	Most evaluation Explore data	Mass evaluation
Small data	Mid sized data	Full sized data

Software needs to be scalable to support different infrastructure

Data management – The FAIR principle

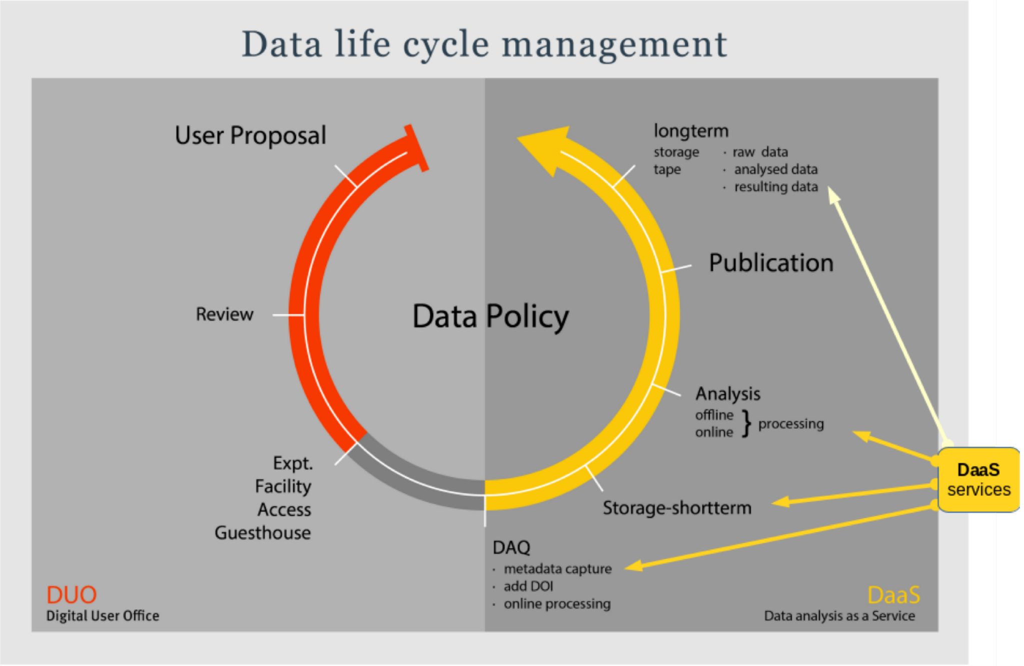
Findable
Accessible
Interoperable
Reproducible

A requirement from funding agencies

- Recording meta data
- Electronic log books

(no more paper and glue)

- Storage
- Data identifiers



- Common and efficient data formats

Wilkinson, Nature Data, 2016, https://doi.org/10.1038/sdata.2016.18

Developing analysis software

Developing software for neutron imaging

Neutron imaging: The final frontier – These are the experiments by many scientists – Their many year mission – To explore strange new worlds – To seek out new samples and new processes

… To boldly go where no man has gone before.

Why?

- Explore new scientific ideas.
- Flexibility and experimental creativity.
- No commercial options available.
- To provide open source alternatives.

How?

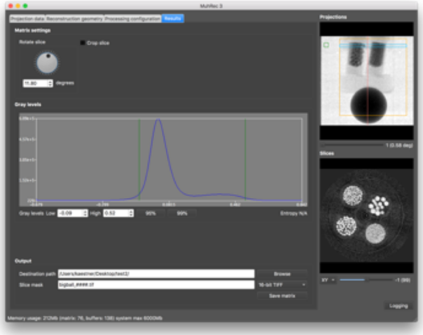
- Scripting languages
- High performance languages
- Existing platforms/from scratch

Applications developed at PSI – White beam

GUI applications are great for specific tasks that require interaction.

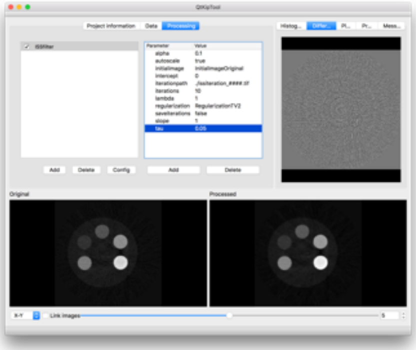
MuhRec

CT reconstruction tool



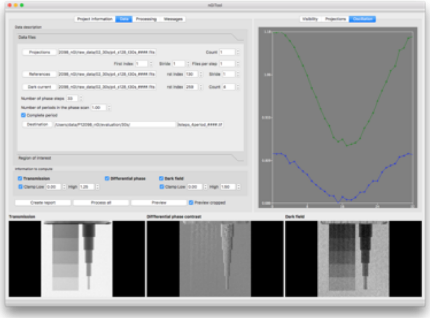
KipTool

General processing tool for 3D images



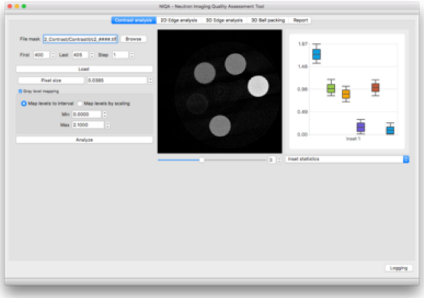
nGI tool

Reduction of phase stepping scans



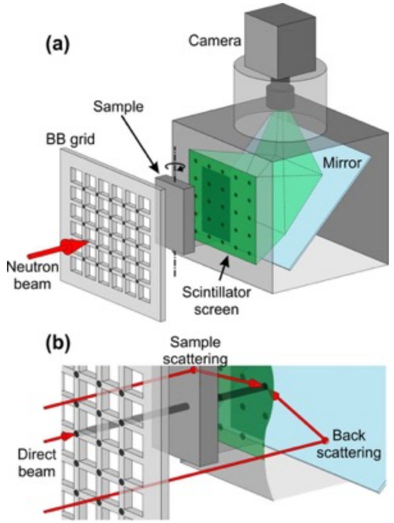
NIQA tool

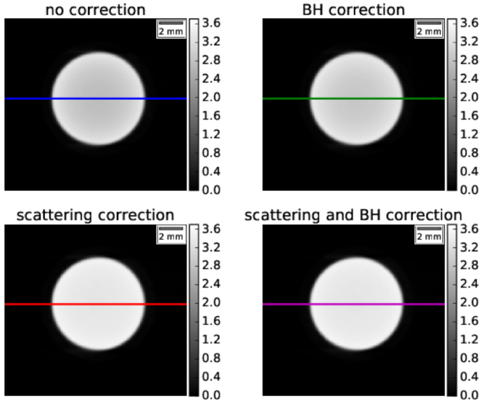
Analysis of IAEA QA samples

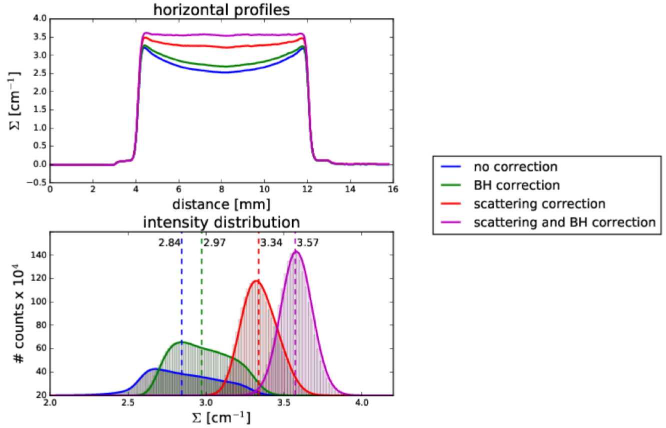


https://neutronimaging.github.io/

The effect of scattering and its correction







Correction using revised normalization

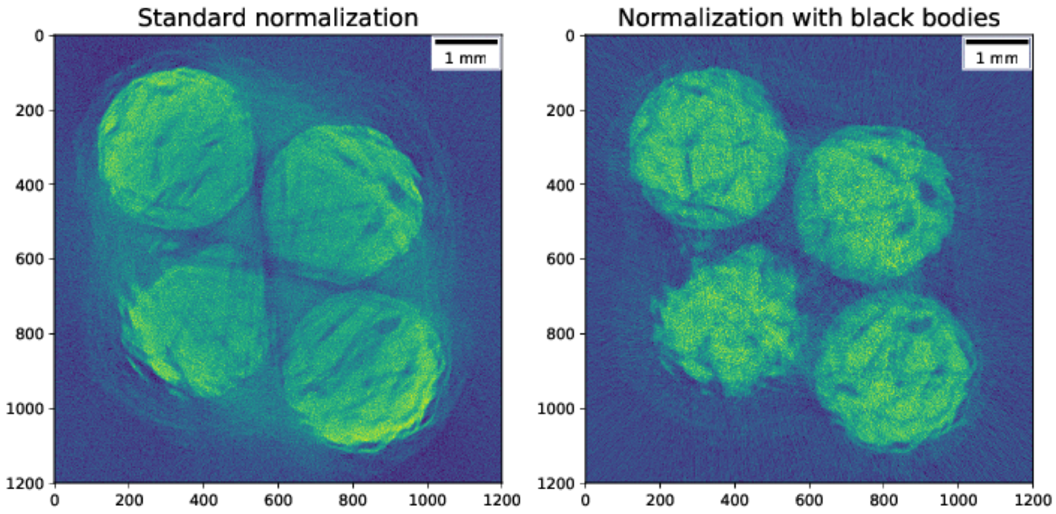
$$\frac{\tilde{I}_n}{\tilde{I}_{OB}} = \frac{I_n}{I_{OB}} \cdot \frac{\mathrm{D}(I_{OB})}{\mathrm{D}(I_n)}$$

⬇

$$\frac{\tilde{I}_n}{\tilde{I}_{OB}} = \frac{I_n}{I_{OB}} \cdot \frac{D(I_{OB})}{D(I_n)}$$

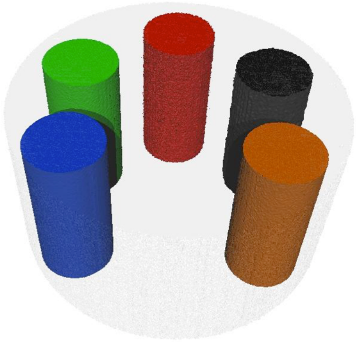
$$= \frac{I_n^* - I_{DC} - I_{n,BB}^S \dfrac{D(I_n^* - I_{DC})}{D\left(I_{n,BB}^* - I_{DC} - \left(1 - \frac{1}{\tau_{BB}}\right) I_{n,BB}^S\right) \tau_{BB}}}{I_{OB}^* - I_{DC} - I_{BG,BB}^S \dfrac{D(I_{OB}^* - I_{DC})}{D\left(I_{OB,BB}^* - I_{DC} - \left(1 - \frac{1}{\tau_{BB}}\right) I_{BG,BB}^S\right) \tau_{BB}}}$$

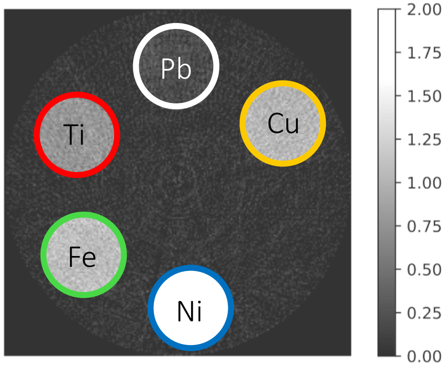
$$\cdot \frac{D\left(I_{OB}^* - I_{DC} - I_{BG,BB}^S \dfrac{D(I_{OB}^* - I_{DC})}{D\left(I_{OB,BB}^* - I_{DC} - \left(1 - \frac{1}{\tau_{BB}}\right) I_{BG,BB}^S\right) \tau_{BB}}\right)}{D\left(I_n^* - I_{DC} - I_{n,BB}^S \dfrac{D(I_n^* - I_{DC})}{D\left(I_{n,BB}^* - I_{DC} - \left(1 - \frac{1}{\tau_{BB}}\right) I_{n,BB}^S\right) \tau_{BB}}\right)}$$

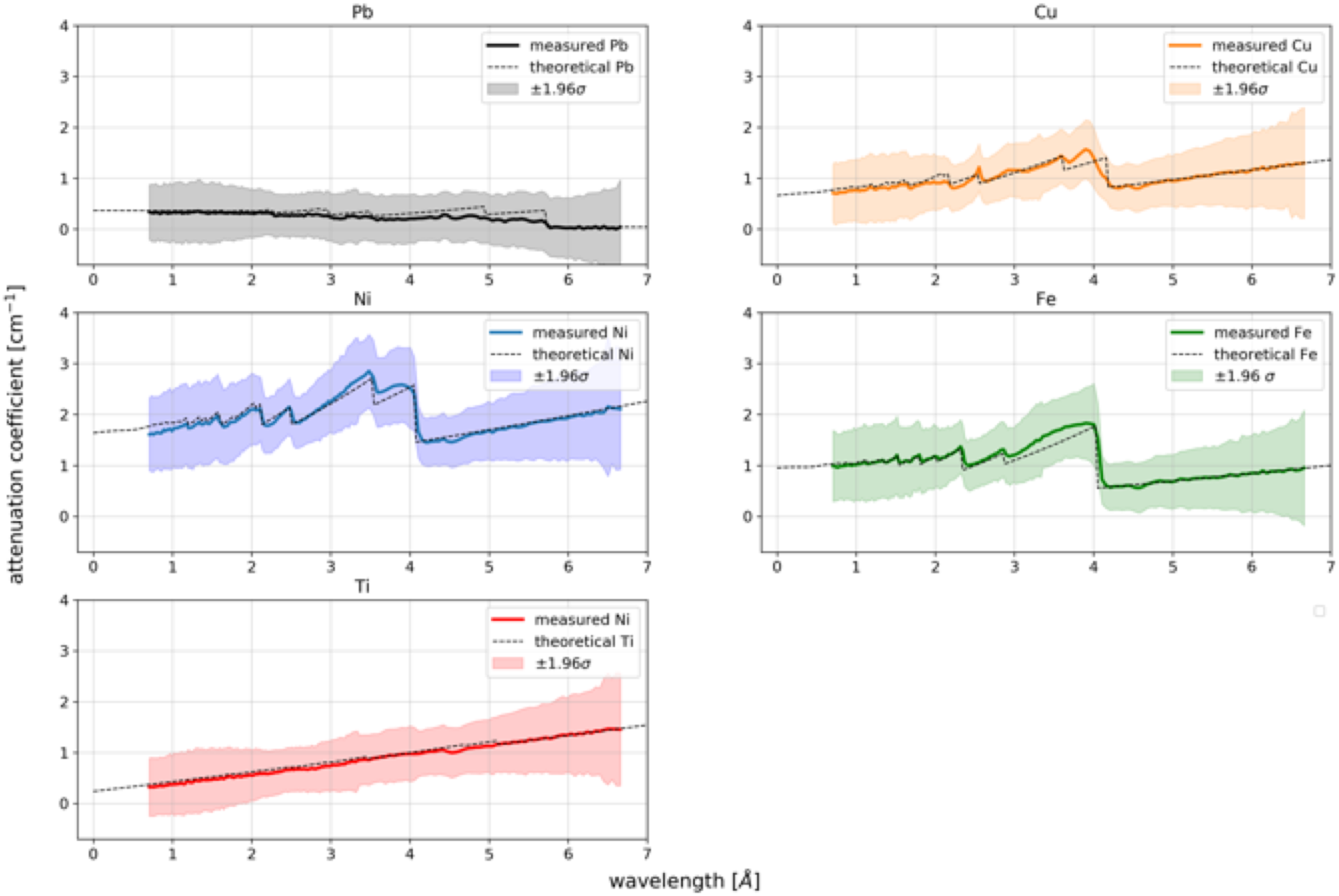


Boillat et al, Opt Express, 2018, Carminati et al, PLOS One, 2019

Energy resolved imaging – ToF Tomography







Carminati et al, J. Applied Crystallography, 2020

Scripting using python

Scripting is the choice for mass production and increased repeatability.

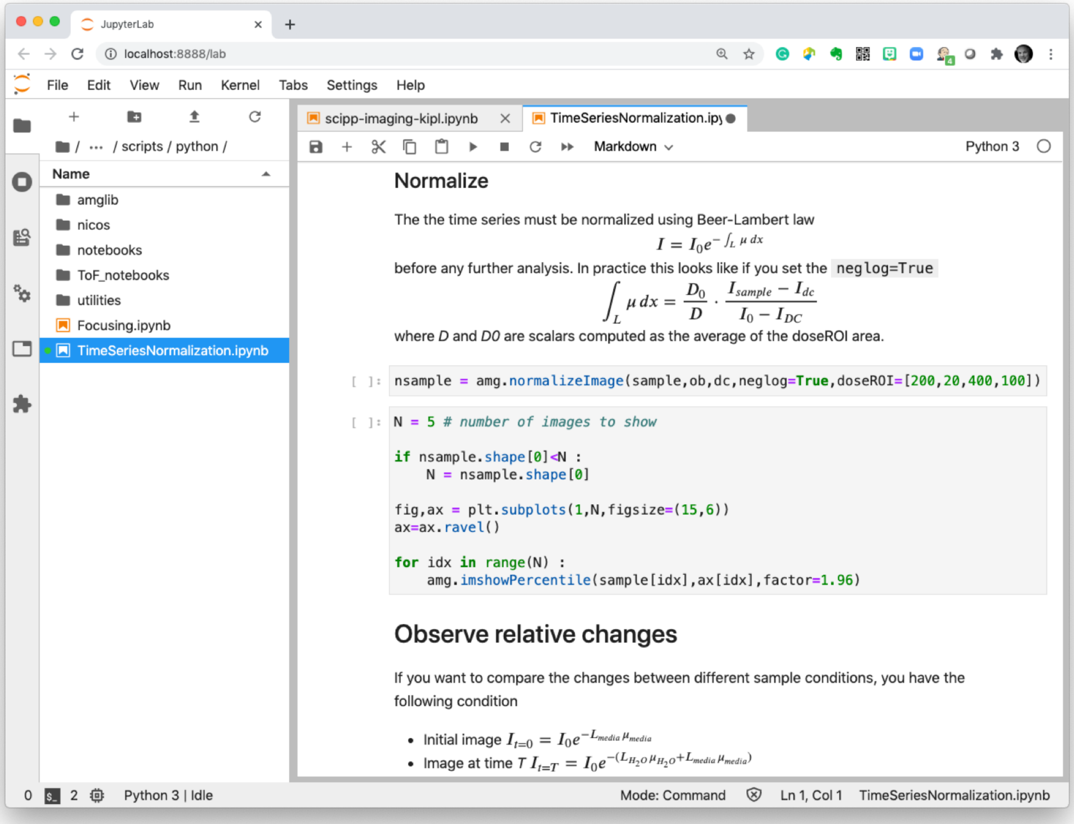
Development focus

- Time-of-flight tool box (Bragg-edge fitting is central)
- Provide workflow notebooks
  - Experiment setup
  - Time series analysis
  - Mass reconstruction
  - …
- Provide an interface to C++ algorithms
  - Spot cleaning
  - Denoising
  - Scattering correction
- Interoperability with multi-dim array module SCIPP (https://scipp.github.io/)

Analysis using Jupyter notebooks

Notebooks

- Provide some interaction
- Are less intimidating than writing pure code
- Allow to document the analysis steps



Normalize

The the time series must be normalized using Beer-Lambert law

$$I = I_0 e^{-\int_L \mu\, dx}$$

before any further analysis. In practice this looks like if you set the `neglog=True`

$$\int_L \mu\, dx = \frac{D_0}{D} \cdot \frac{I_{sample} - I_{dc}}{I_0 - I_{DC}}$$

where *D* and *D0* are scalars computed as the average of the doseROI area.

```
[ ]: nsample = amg.normalizeImage(sample,ob,dc,neglog=True,doseROI=[200,20,400,100])
```

```
[ ]: N = 5 # number of images to show

     if nsample.shape[0]<N :
         N = nsample.shape[0]

     fig,ax = plt.subplots(1,N,figsize=(15,6))
     ax=ax.ravel()

     for idx in range(N) :
         amg.imshowPercentile(sample[idx],ax[idx],factor=1.96)
```

Observe relative changes

If you want to compare the changes between different sample conditions, you have the following condition

- Initial image $I_{t=0} = I_0 e^{-L_{media}\mu_{media}}$
- Image at time T $I_{t=T} = I_0 e^{-(L_{H_2O}\mu_{H_2O}+L_{media}\mu_{media})}$

Modern software development

- Agile development
  - Iterative development
  - Collaboration
- Using repositories
  - New issues in repository branches
  - Maintain stable master through merge reviews
- Issue tracking (New features, Improvements, Fixing bugs)
- Frequent releases (2 x year)
- Automated testing and builds
- Online documentation (wiki)

Some technical details about the development

Main coding language

In particular: C++11
- XCode (Mac)
- MSVC15 (Windows)
- g++ (Linux)

GUI Library

Version: 5.15 LTS
- Cross platform
- Qmake/CMake
- Qtest

Python bindings

Python 3 + PyBind11
- *Embedded scripting*
- Call libs from python
- *Tutorials*
- *Typical work flows*

Repository: GitHub

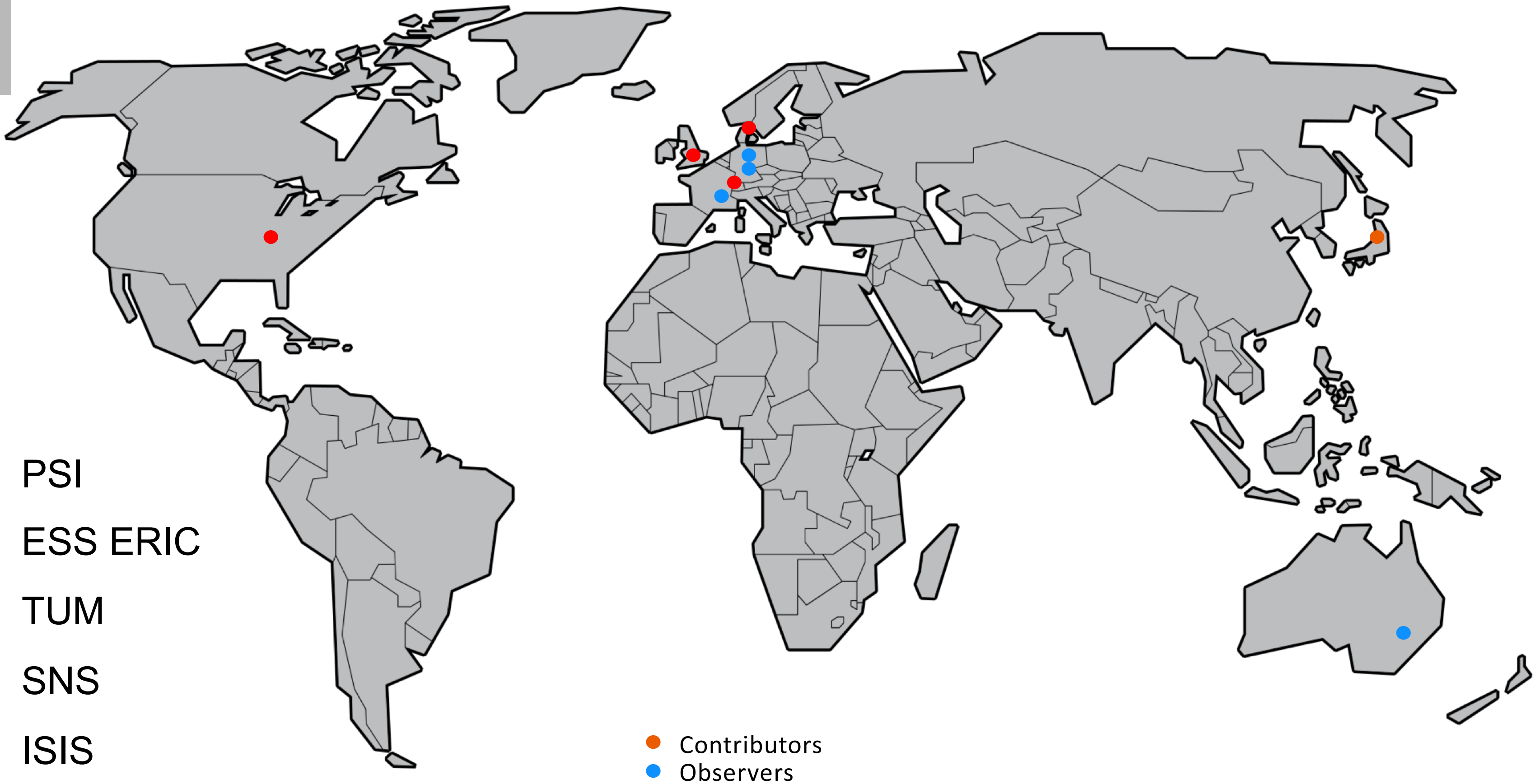
https://github.com/neutronimaging

- Development history
- Issue tracking
- Documentation (Wiki)

Build server: Jenkins

- Automated builds
- Nightly build installers

Collaboration Network



PSI

ESS ERIC

TUM

SNS

ISIS

J-PARC

Recent new collaboration efforts with TUM and ILL

The analysis of imaging data should be …

… done in a reproducible way.

… open source to promote collaboration.

… developed with modern techniques.

We welcome new partners to join.

https://neutronimaging.github.io/

Acknowledgements

Development team

- Chiara Carminati
- Matteo Busi

User feedback

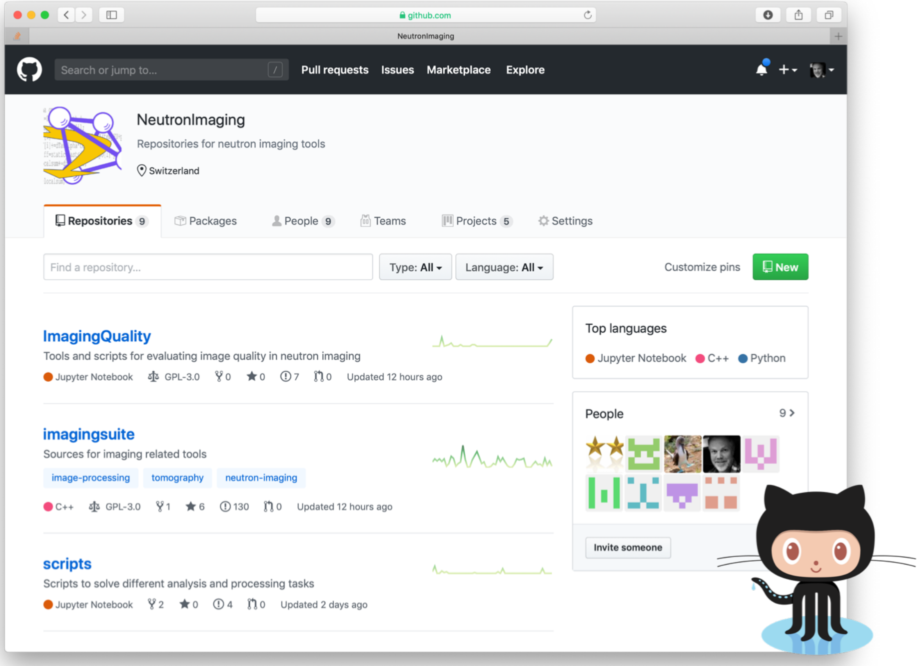
- AMG
- Experiment users

Funding

ESS - DMSC

Open source and distribution - Repository

All developed code shall be open source



NeutronImaging

Repositories for neutron imaging tools

Switzerland

ImagingQuality

Tools and scripts for evaluating image quality in neutron imaging

Jupyter Notebook GPL-3.0 0 ★ 0 7 0 Updated 12 hours ago

imagingsuite

Sources for imaging related tools

image-processing tomography neutron-imaging

C++ GPL-3.0 1 ★ 6 130 0 Updated 12 hours ago

scripts

Scripts to solve different analysis and processing tasks

Jupyter Notebook 2 ★ 0 4 0 Updated 2 days ago

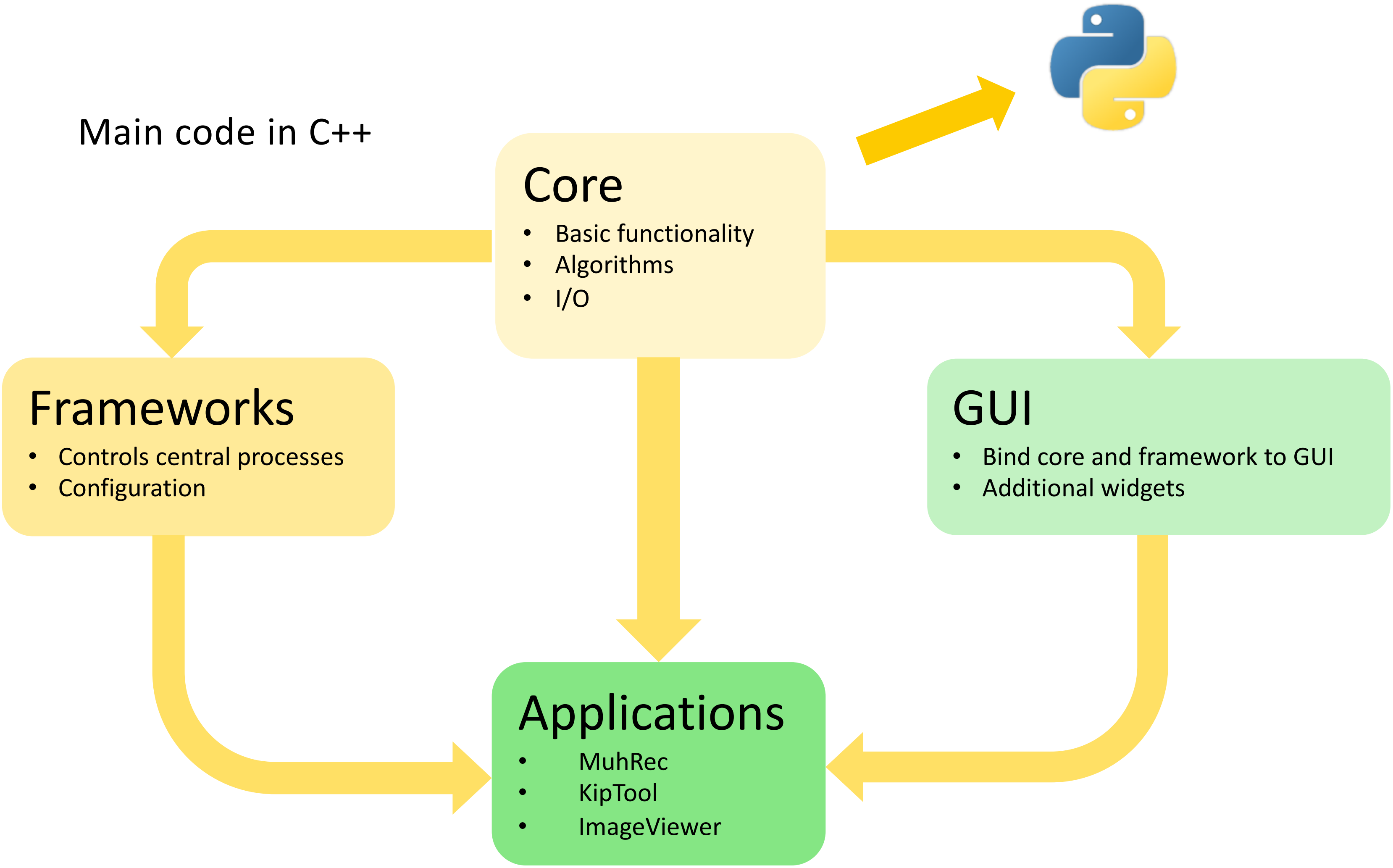
Done

- Most of the code is transferred
- License GPL 3.0
- Issue tracking (GitHub)
- Submit issue link in apps
- Transfer remaining code
- Prepare build scripts
- Setup automated build
- Prepare build instructions
- Documentation on Wiki

Source repository https://github.com/neutronimaging

Design – Strict separation per purpose

Main code in C++



Core

- Basic functionality
- Algorithms
- I/O

Frameworks

- Controls central processes
- Configuration

GUI

- Bind core and framework to GUI
- Additional widgets

Applications

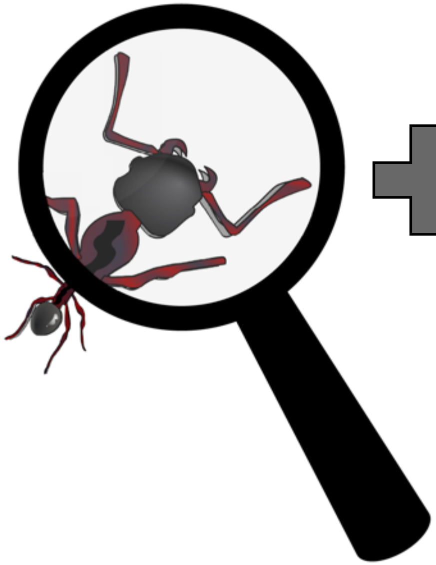
- MuhRec
- KipTool
- ImageViewer

The challenge

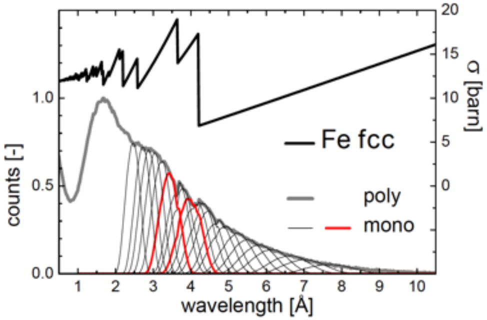
Our experiment users want …

Fast acquisition

High resolution



Monochromatic



But the number of neutrons is limited…

Characterizing neutron imaging data

Data quality

Limited number of observations	*long experiment times*
Low SNR	*few detected neutrons per pixel*
Less sharp than X-rays	*detection principle*

Data structure

Single modality	*traditional transmission imaging*
Temporal dependency	*time series*
Multi modality	*combining images from different sources*
Multi wavelength	*material response, mixes*
Tensor valued	*magnetism, strain, diffraction*

Investigations often only involve few or even single samples …
Is generalization possible?