ESRF | The European Synchrotron

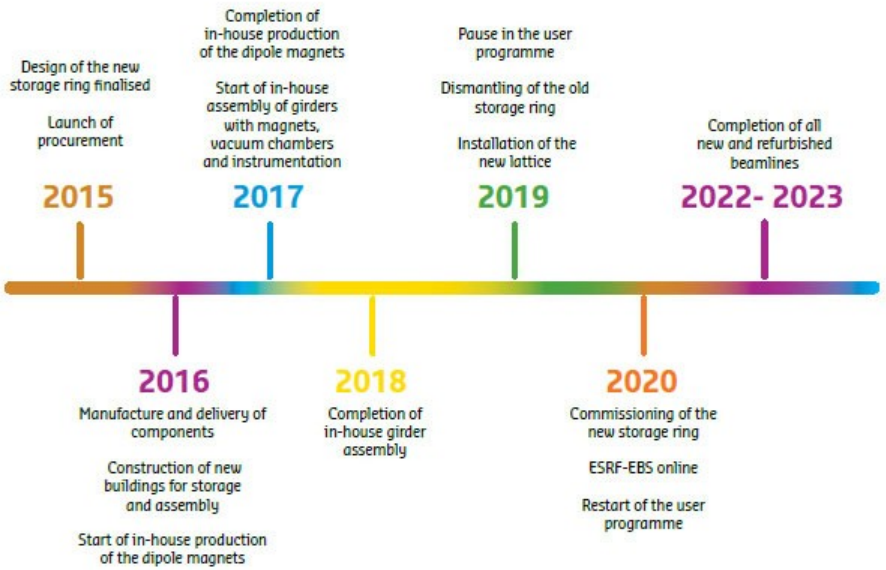
The new tomography data processing software of the ESRF

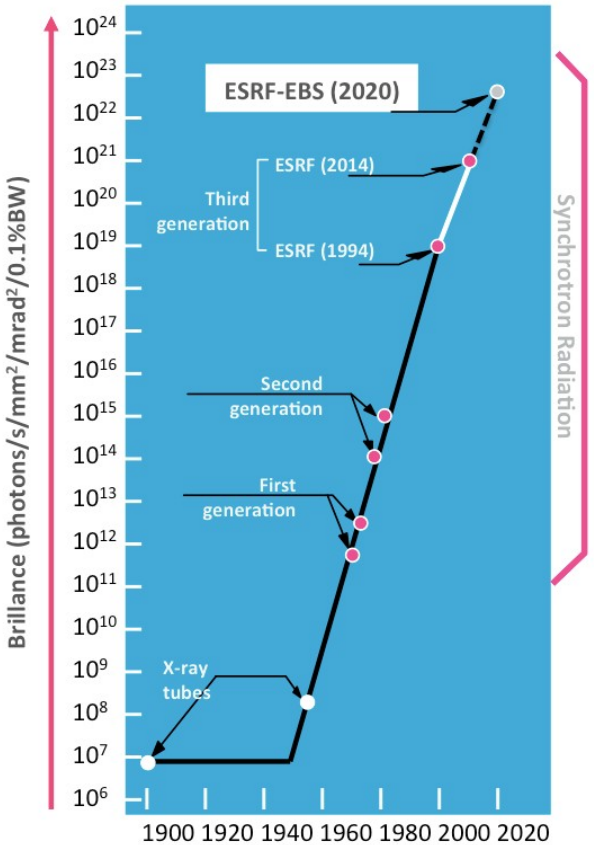
Outline

- Why new data processing software for tomography ?
- Design / layout of the new data processing software suite
- Software details
  - NXtomomill
  - nabu
  - tomwer
- Future

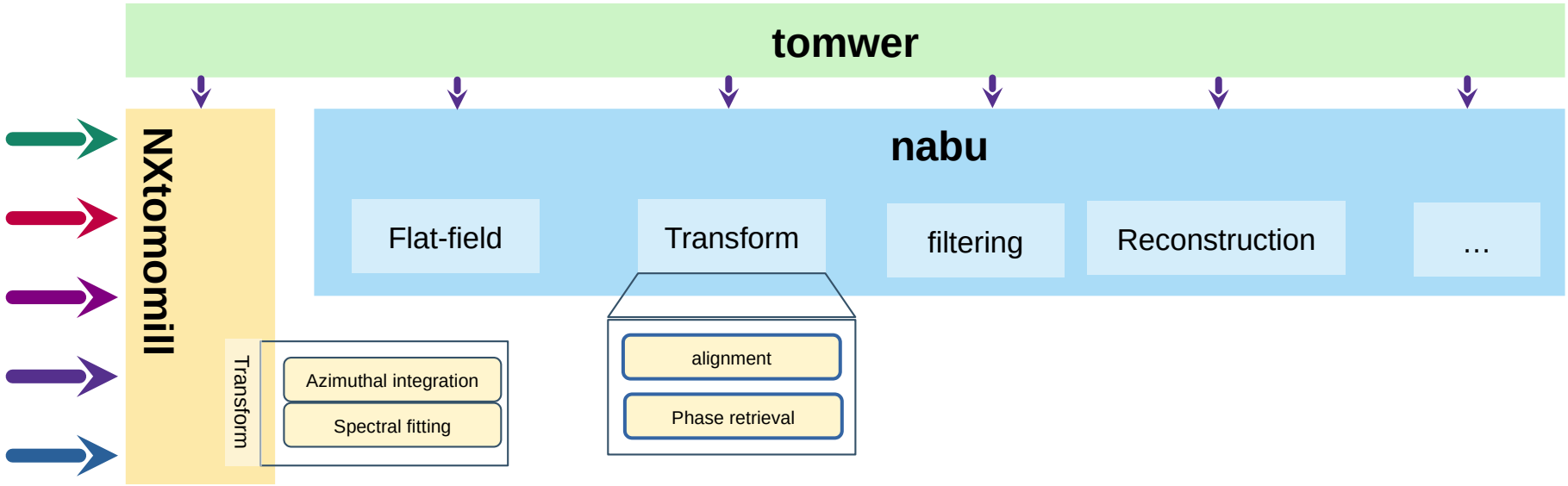
Why new data processing software for tomography ?

- EBS upgrade
- Existing software at the time
  - Pyhst2 for the reconstruction
  - Spec for the acquisition
  - scripts spread across different beamlines
    - Example: fastomo for preprocessing





Design / layout of the new data processing software suite

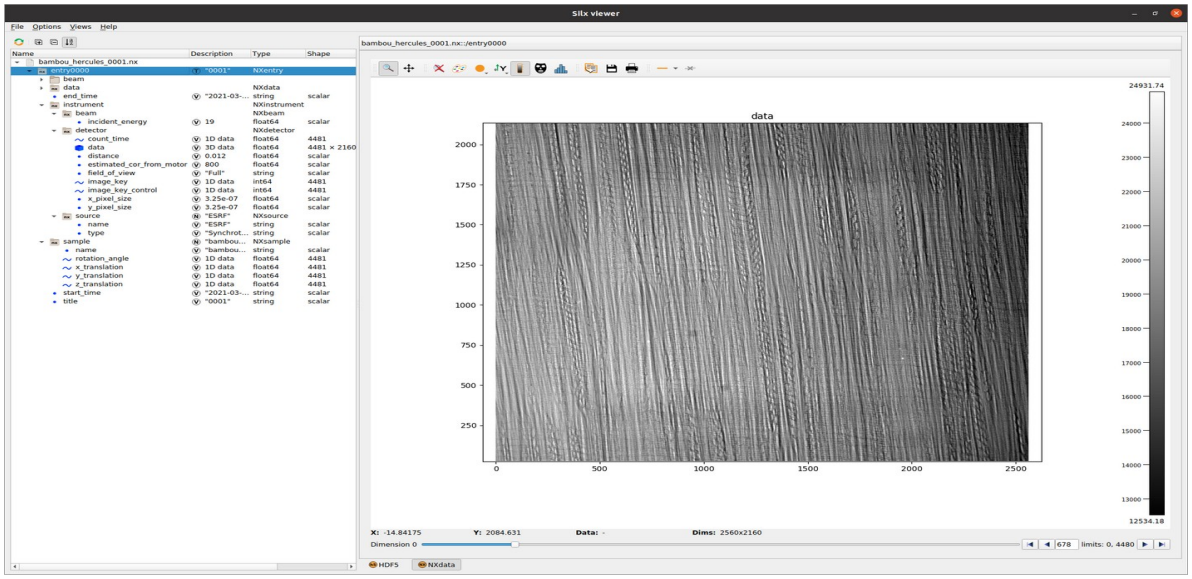


- **nxtomomill**: data conversion / reduction
- **nabu**: tomography processing library
- **tomwer**: workflow definition and GUI

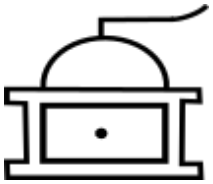
NXtomomill: NXtomo creator since 2020

- **insure a stable format**

- An NXtomo contains all the information necessary to process tomography data. frames (dark, flat, projections...), rotation angles, x, y and z translations, etc…



silx view displaying an NXtomo from `bambou_hercules` dataset



pip install nxtomomill

NXtomomill: context

- Acquisitions are now done by bliss
  - Saved as a generic nexus hdf5
  - Layout / structure can be diversify from one beamline and/or acquisition to the other
- Backward compatibility with spec-EDF
- data Exchange file format also supported
- Conversion can be done from the python API or the CLI

```python
from nxtomomill.io.config import TomoHDF5Config
from nxtomomill.converter.hdf5.hdf5converter import from_h5_to_nx

configuration = TomoHDF5Config()
configuration.input_file = "bliss_file.hdf5"
configuration.output_file = "nexus_file.nx"
from_h5_to_nx(configuration)
```

Python API

```
nxtomomill dxfile2nx tomo_00068.h5 tomo_00068.nx
```

```
nxtomomill h52nx bliss.hdf5 my_nxtomo_file.nx
```

CLI

- Include a `nexus` module to ease definition and edition of an NXtomo from python

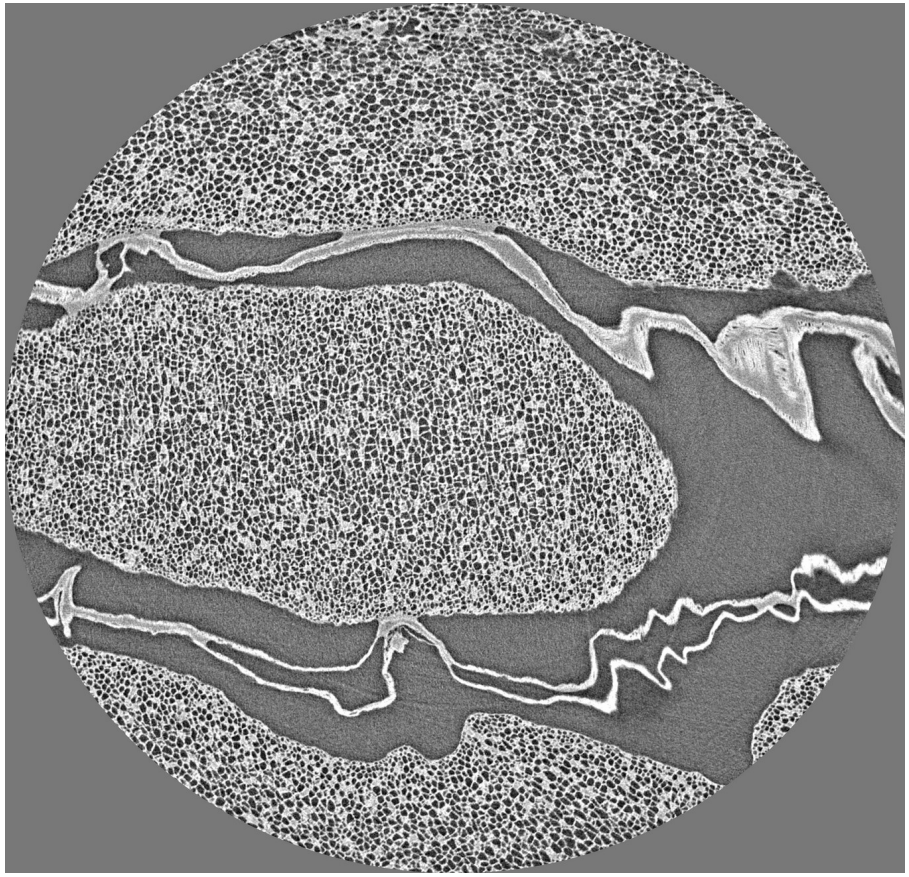
nabu: high performance tomographic reconstruction software

Nabu is a **versatile tomography reconstruction** software.

- Tailored to ESRF needs
  - Many features not available in existing tomography software
  - Reconstruction is only a fraction of the overall pipeline !
  - Support legacy formats/methods
  - Adapt quickly to users needs and ideas

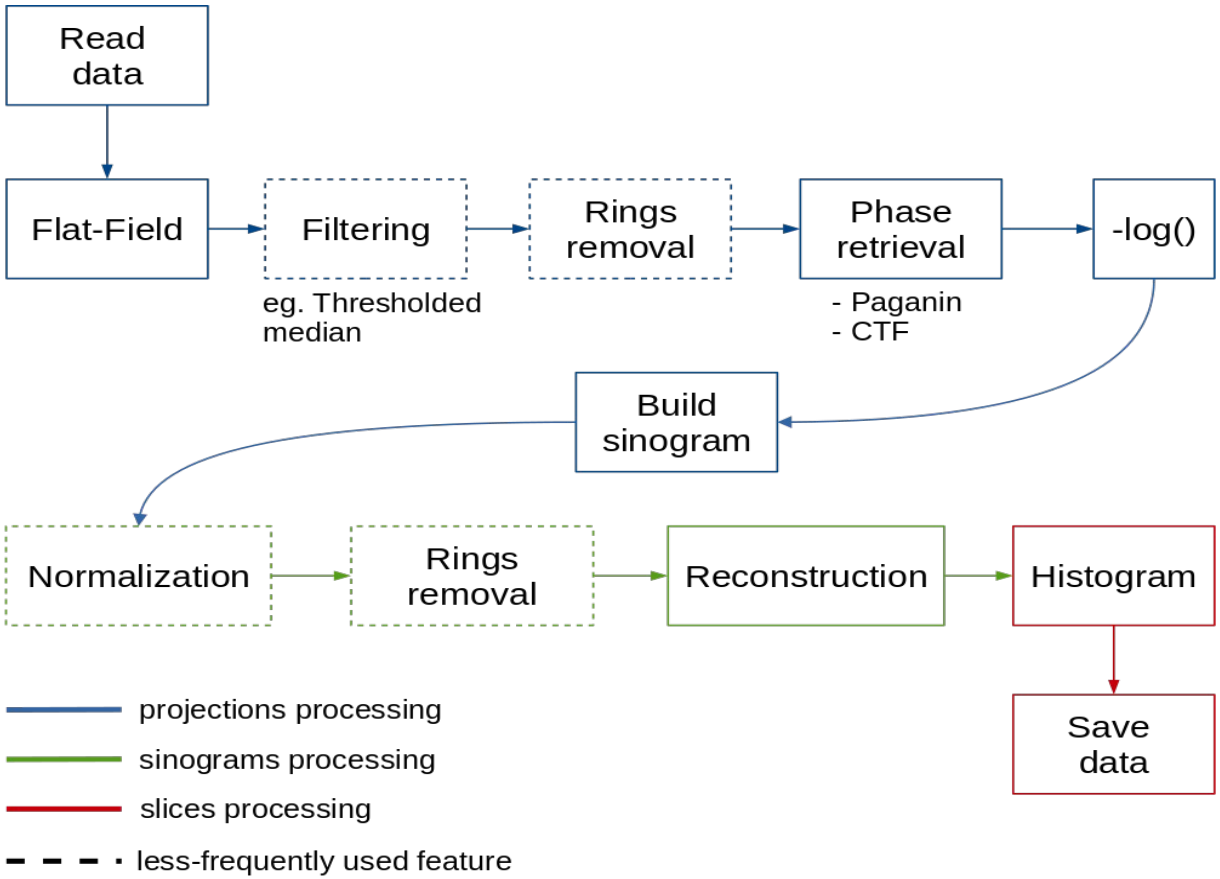
    → need control over the code-base
- Set of **high performance** building blocks

- On top of them, **reconstruction pipelines** are built for specific use cases:
  - Fast full-field reconstruction → now (absorption/phase contrast)
  - Diffraction tomography → work in progress
  - Fluorescence tomography → in the future



Coffe bean – ID19

nabu: high performance tomographic reconstruction software



Full field reconstruction pipeline

- High speed. **All processing steps are done on GPU**.
  - CPU backend is available
- Estimation tools
  - Alignment, center-of-rotation, overlap for phase retrieval, ...
- Save **checkpoints** in the processing, resume from a saved

nabu: processing classes example

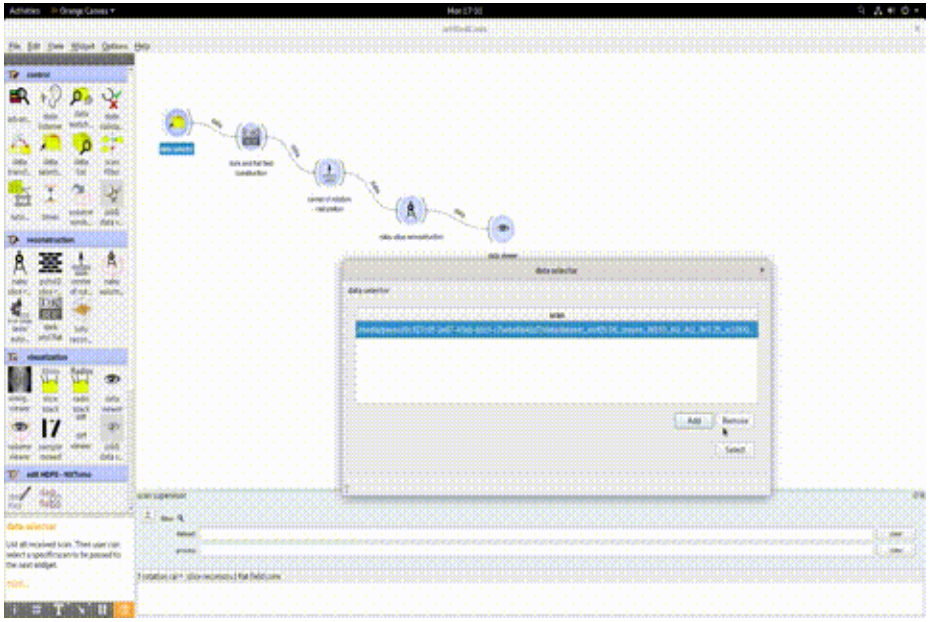
- Nabu classes and functions act on arrays directly (numpy, pycuda)
  - Low barrier to entry
  - Focus on our “core business”: tomography

```python

# ...imports

data_path = "/path/to/my/dataset.nx"
dataset_info = analyze_dataset(data_path)

# Read the first 100 lines of all projections
reader = ChunkReader(
    dataset_info.projections, sub_region=(None, None, 0, 100), convert_float=True
)
reader.load_data()
n_angles, n_z, n_x = reader.chunk_shape
projections = reader.data

# Initialize processing classes
flatfielder = FlatField(
    reader.chunk_shape,
    dataset_info.flats,
    dataset_info.darks,
    radios_indices=list(dataset_info.projections.keys()),
    sub_region=reader.sub_region,
)
phase_retrieval = PaganinPhaseRetrieval(
    (n_z, n_x), distance=0.2, energy=19, delta_beta=100, pixel_size=2.5 * 1e-6
)
fbp = Backprojector((n_angles, n_x), angles=dataset_info.rotation_angles)

reconstructions = np.zeros((n_z,) + fbp.slice_shape, dtype="f")

# Process data
flatfielder.normalize_radios(projections)
for i_angle in range(n_angles):
    phase_retrieval.apply_filter(projections[i_angle], output=projections[i_angle])
for i_slice in range(n_z):
    reconstructions[i_slice] = fbp.fbp(projections[:, i_slice, :])
```

pip install nabu

tomwer: tomography workflow

- tomwer goal: ease tomographic data processing.
  - `tomwer canvas`: let users define processing workflow(s)
    - from acquisition to reconstructed volume
    - Ensure each steps can be ‘connected’
  - If a picture is sometimes worth a thousand words what about a video ?
    - Presentation video
    - video tutorials

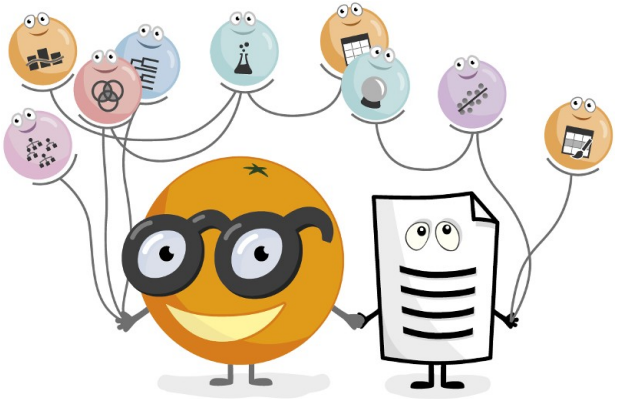


pip install tomwer[full]

tomwer: tomography workflow

More features:

- NXtomo edition
- Job slurm submission
- Insert user python script
- Workflows triggered by bliss
- …



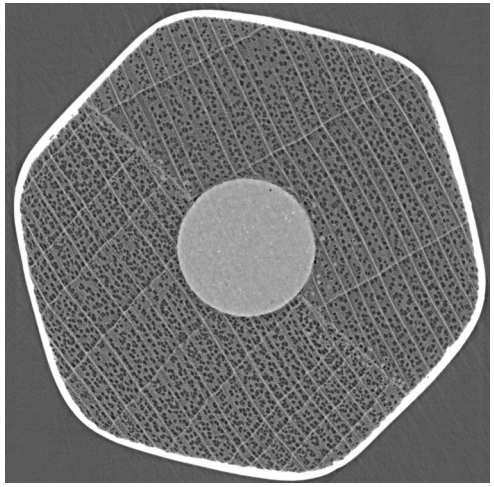
Rely on several software

- silx
- ewoks
- Orange-canvas-core and orange-widget-base
- h5py
- …

Future

- More techniques ‘under the same umbrella’
  - XRF-CT, multi-phase …
- Allow more ‘call’ / integration of external software
- Support for more complex data (pre and post processing)
  - stitching projections, reconstructed volumes…
- Integrating Machine Learning ?
- Acknowledgments
  - contributors: N. Vigano, C. Nemoz, A. Mirone, V. Favre-Nicolin, Y. Meyer, A. Sole, P-J Gouttenoire
  - friendly users: E. Boller, B. Cordonnier, L.Broche, M. Majkut, P-O. Autran

- More man power
  - https://esrf.gestmax.eu/1605/1/scientist-x-ray-tomography/en_US
  - https://esrf.gestmax.eu/1604/1/scientist-coherent-x-ray-imaging/en_US



pencil – old spec acquisition

Questions ?

https://gitlab.esrf.fr/tomotools/nxtomomill
https://gitlab.esrf.fr/tomotools/nabu
https://gitlab.esrf.fr/tomotools/tomwer