

# Welcome to the QIM + CIL (Core Imaging Library) Reconstruction Workshop: Understanding the impact on image analysis

Grab some post-its and, on separate post-its, write:

1. If you could scan *anything*, what would it be?  
(be creative!)
2. What do you want to get out of this workshop?  
(your goals, questions, skills to learn)

Keep your post-its, we will talk about them later!



# Photo permission

- We take photos mainly for our website
- Let us know if you do not want to appear in photos

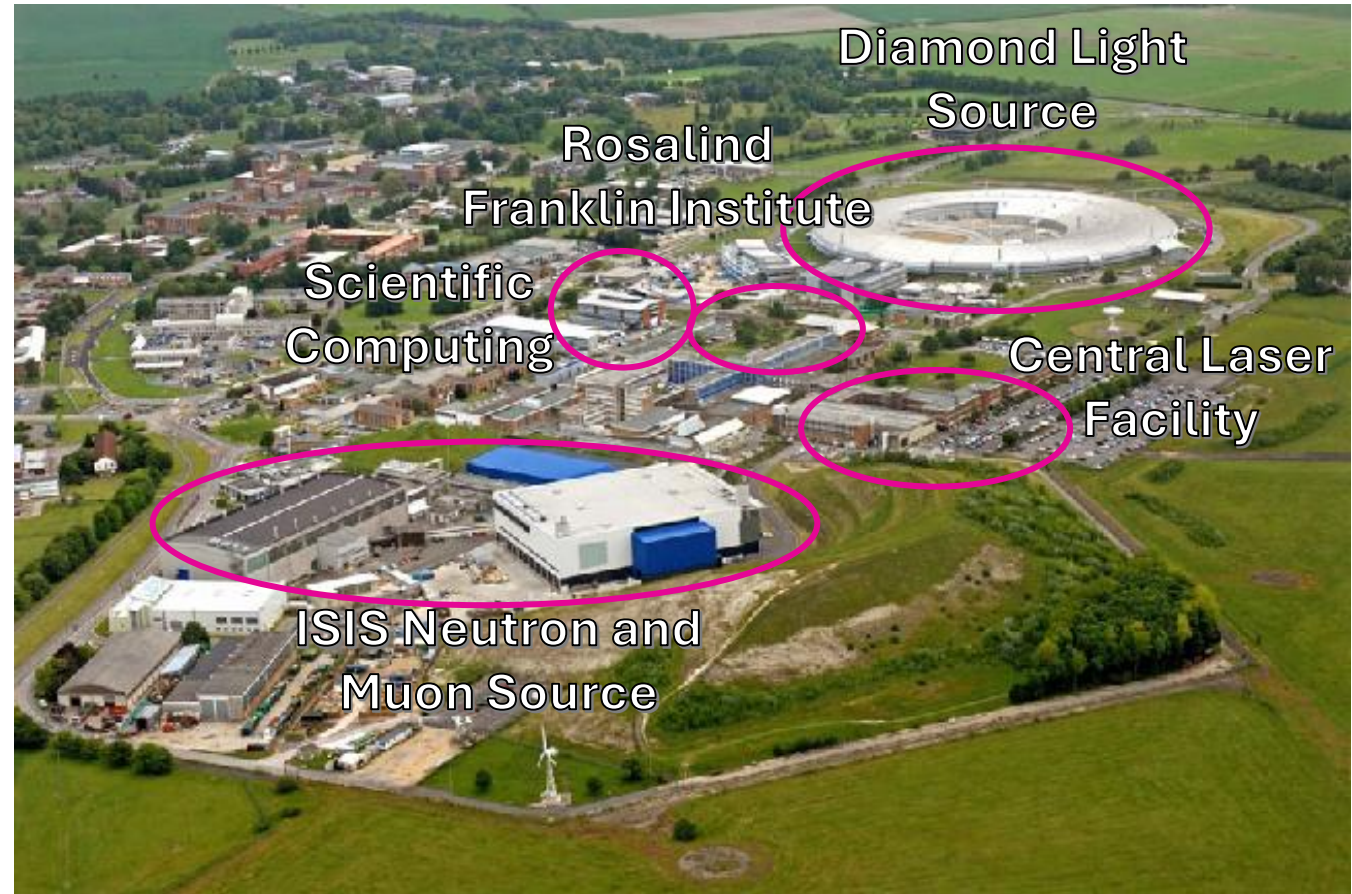
# Reproducible Tomographic Image Processing with the Core Imaging Library

Jakob Sauer Jørgensen, Edoardo Pasca and  
Gemma Fardell

Scientific Computing,  
Science and Technology Facilities Council, UK

# Science and Technology Facilities Council

- Run facilities at the Rutherford Appleton Laboratory, UK
- Scientific Computing provides computing infrastructure, software, support and expertise
- CIL team also at DTU and Finden





# The CIL Team



Casper  
Da Costa-Luis<sup>1</sup>

Danica  
Sugic<sup>1</sup>

Gemma  
Fardell<sup>1</sup>

Jakob  
Sauer Jørgensen<sup>2</sup>

Edoardo  
Pasca<sup>1</sup>

Evangelos  
Papoutsellis<sup>3</sup>

Hannah  
Robarts<sup>1</sup>

Margaret  
Duff<sup>1</sup>

Laura  
Murgatroyd<sup>1</sup>

Franck  
Vidal<sup>1</sup>

1. Scientific Computing @ STFC
2. Technical University of Denmark (DTU)
3. Finden

# STFC Tomography and Imaging

- CCPi: Bring together expertise in computational research for tomography and imaging
  - Open-source software development, maintenance and distribution
  - Methods development
  - Community building
  - Training and user support network



# Goals for this workshop

- Introduce tomography and the Core Imaging Library (CIL)
  - Standard reconstruction methods
  - Data processing in CIL
  - Iterative reconstruction methods
- Show example tailored reconstruction methods on challenging data
- Get you ready to use CIL on your own data

# Introductions

1. If you could scan *anything*, what would it be?  
(*be creative!*)
2. What do you want to get out of this workshop?  
(*your goals, questions, skills to learn*)



# Training plan

## **Session 1 – Welcome, reading and pre-processing (1 hr 30 mins)**

- Welcome and introductions (10mins)
- Introduction to CIL (15mins)
- Get started with the cloud and ESRF pipeline example (20mins)
- Review of ESRF reconstruction pipeline (10mins)
- Exercises – preprocessing or construction your own reader (25mins)
- Wrap up of first session (10mins)

*Break (30mins)*

## **Session 2 – introduction to iterative reconstruction methods (1 hr 15 mins)**

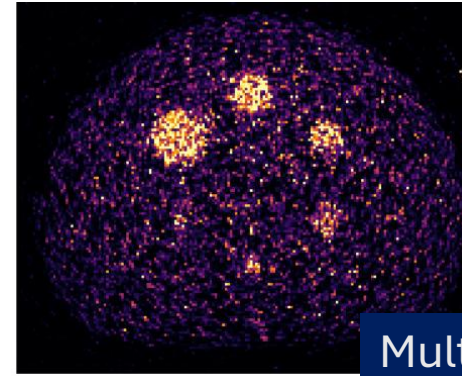
- Intro to iterative reconstruction methods (10mins)
- Example limited angle notebook (20mins)
- Review of the limited angle notebook (10mins)
- Intro to regularisation (10mins)
- Notebooks to explore regularisation (25mins)

## **Final questions, discussions and next steps (15mins)**

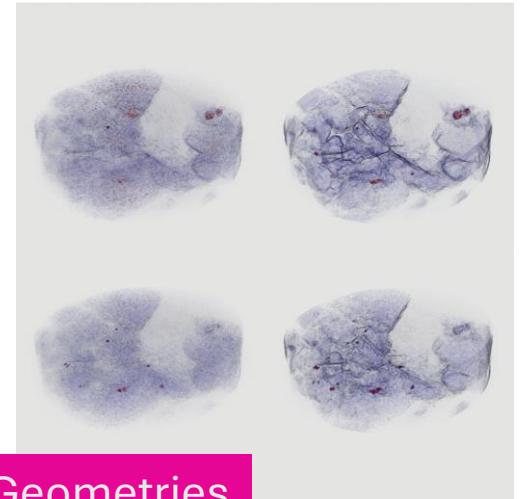
# Core Imaging Library (CIL)

- Open-source python framework for CT
- Methods for cone and parallel beam geometries
- Tools for reading, processing, reconstructing and visualising CT data
- Optimisation tools for iterative reconstruction methods with a focus on challenging data

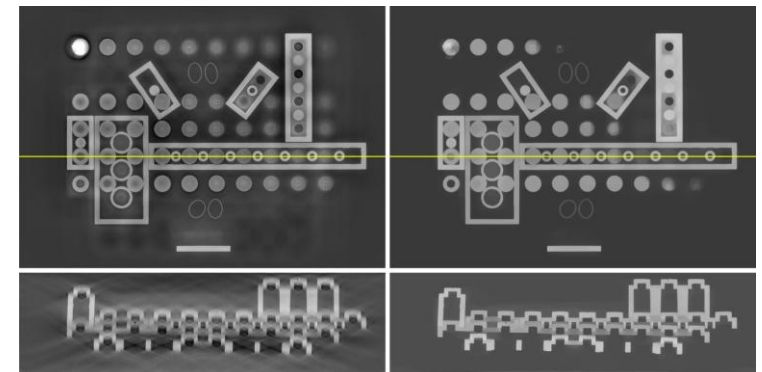
Noisy



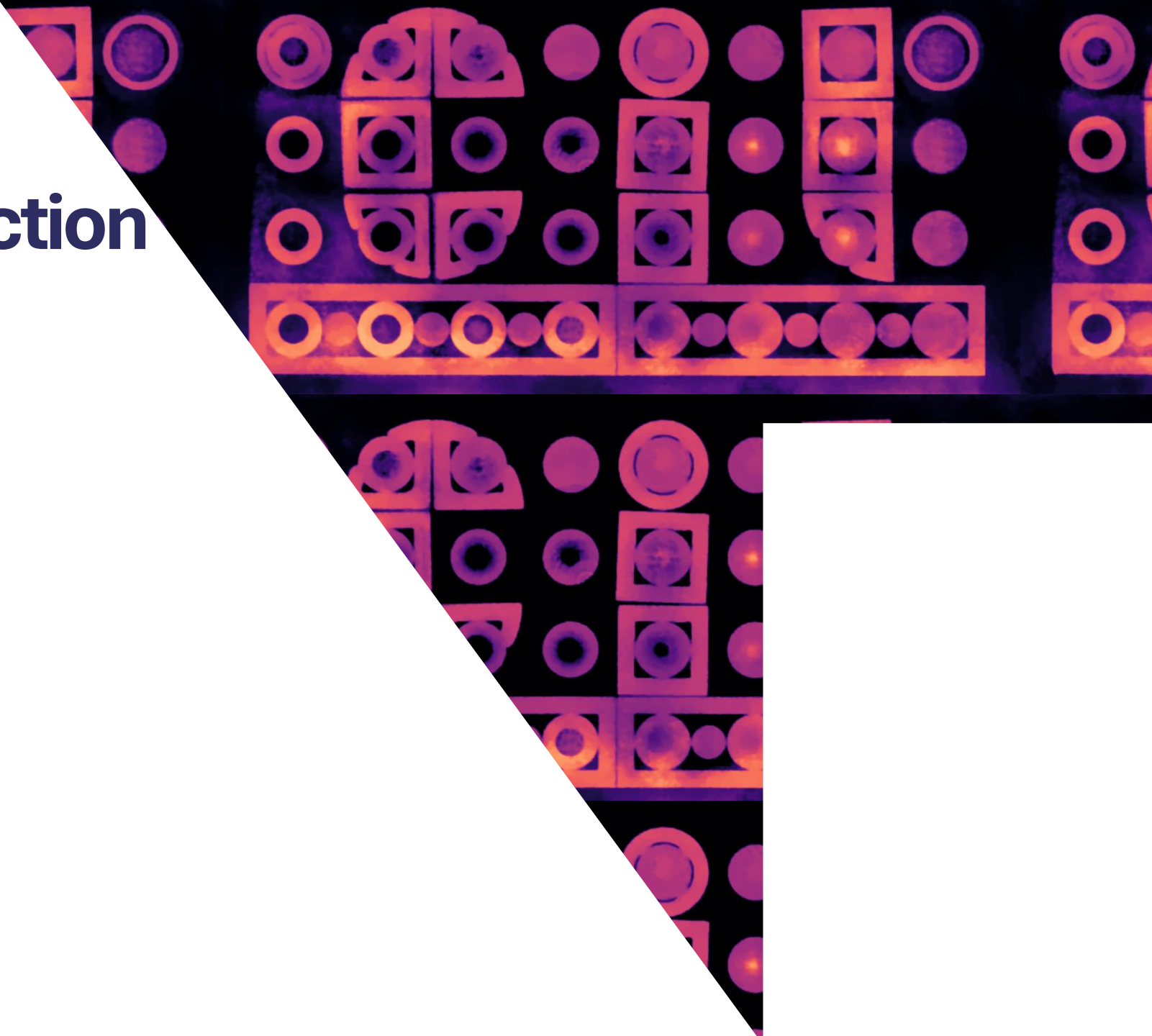
Multi-channel



Non-standard Geometries



# Filtered Back Projection Reconstruction



# Tomography

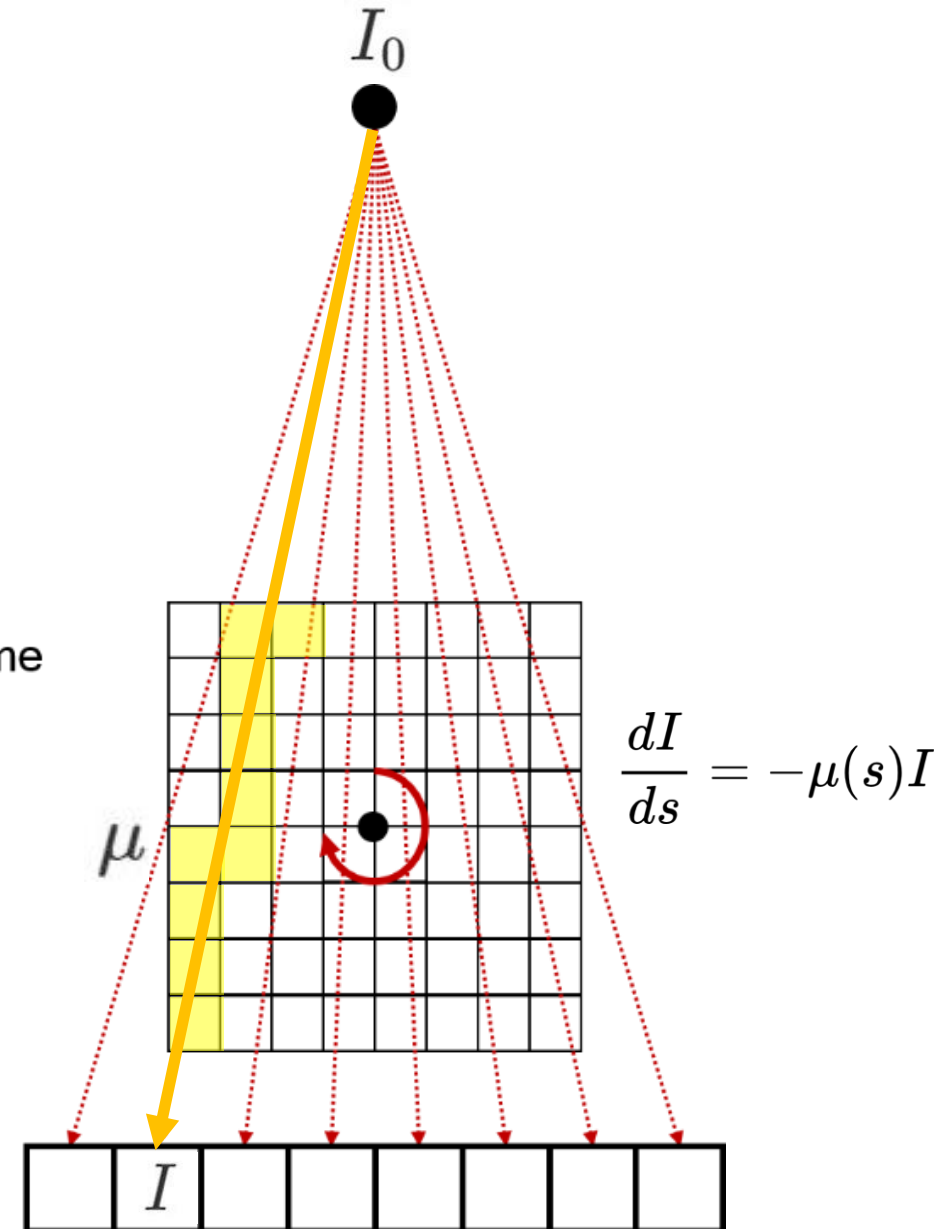
$$\frac{I}{I_0} = \exp \int_{L_i} -\mu(s) ds$$

$$b_i = -\log \frac{I_i}{I_0} = \int_{L_i} \mu(s) ds$$

X-ray source

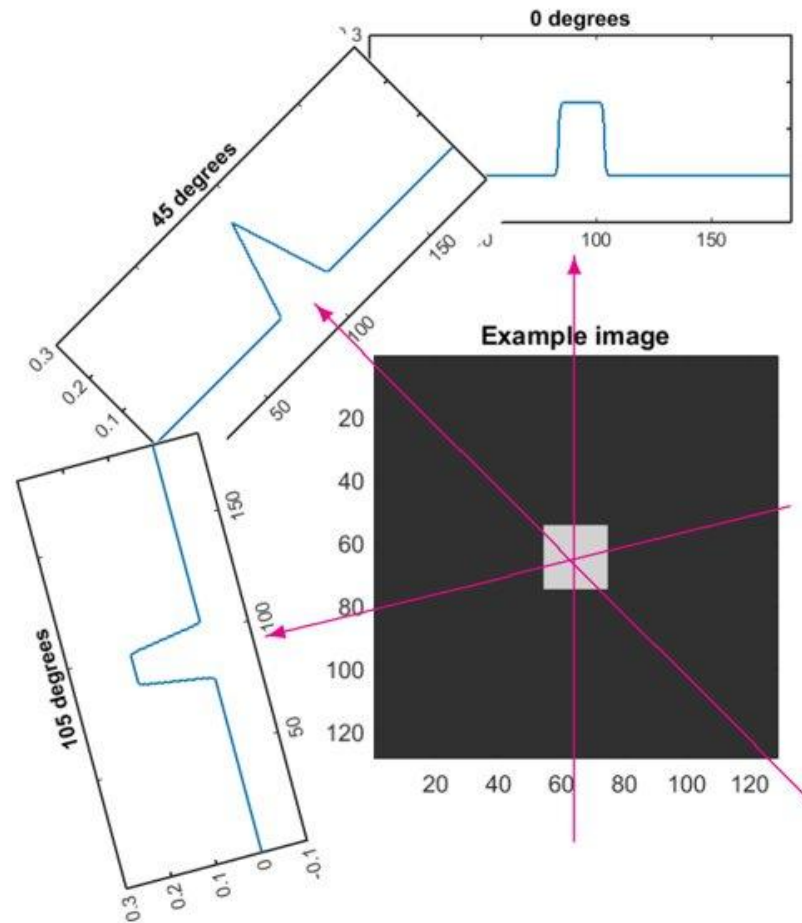
Measurement volume

X-ray detector

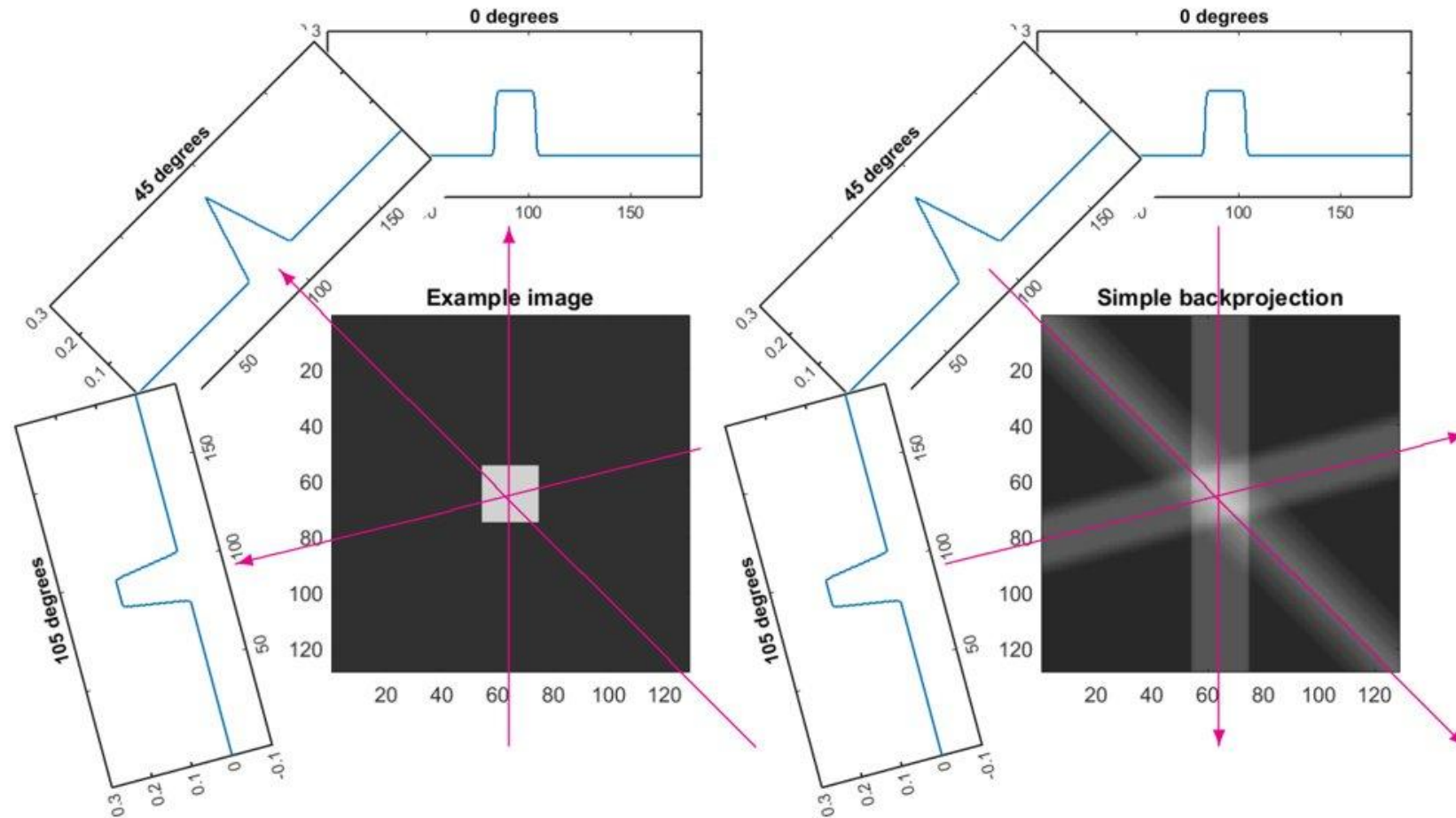




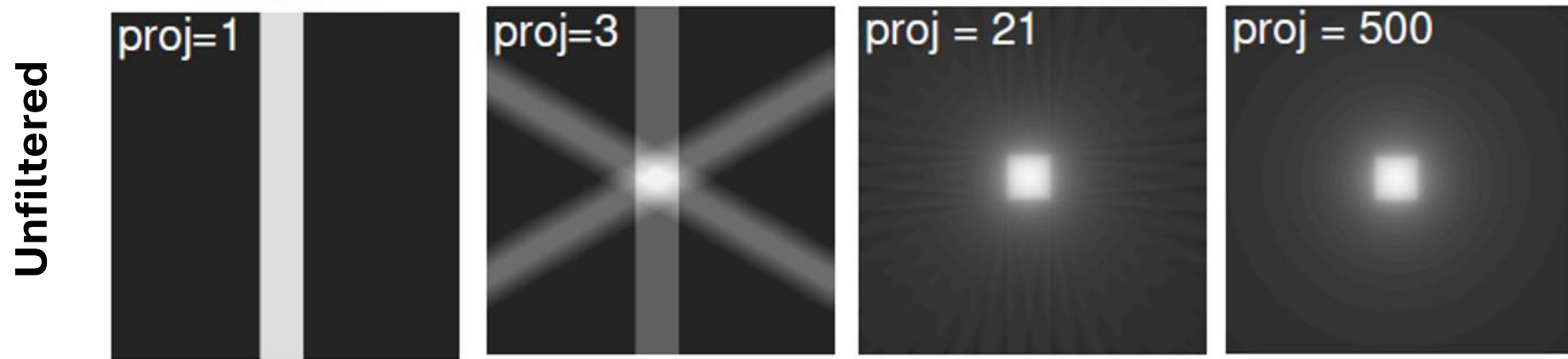
# Filtered Back Projection (FBP)



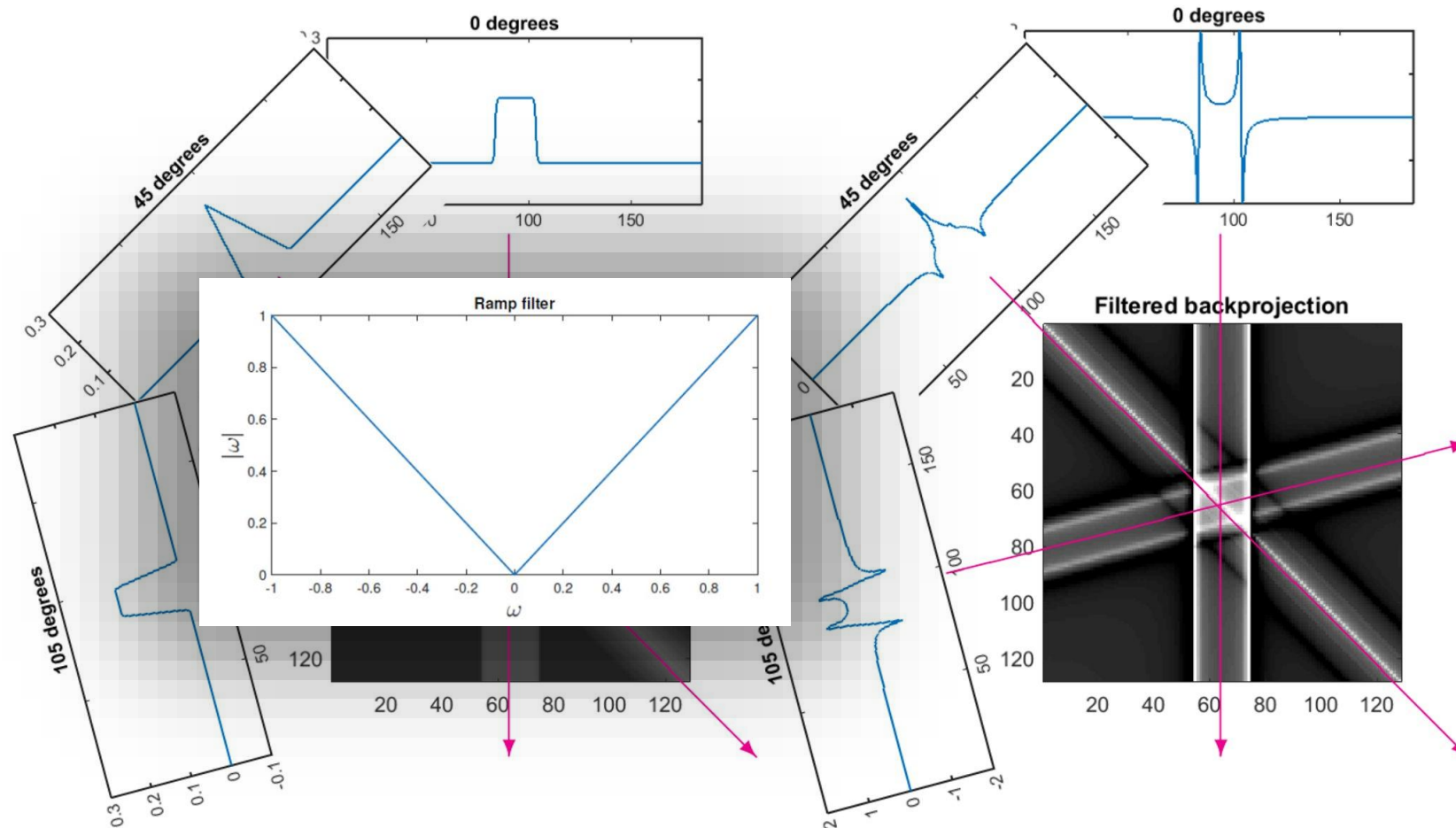
# Filtered Back Projection (FBP)



# Filtered Back Projection (FBP)

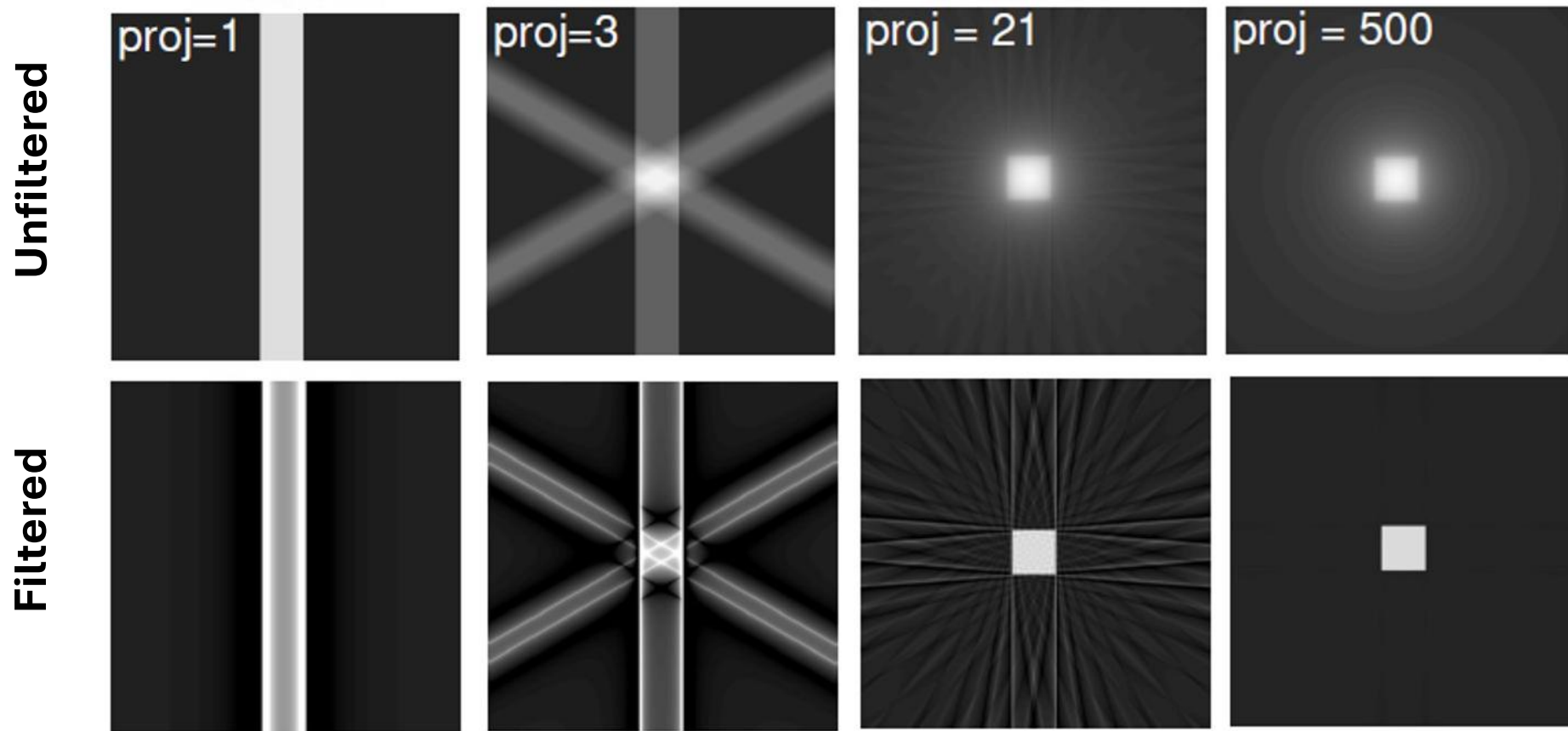


# Filtered Back Projection (FBP)





# Filtered Back Projection (FBP)



# Filtered Back Projection (FBP)

## Pros

- Fast as based on FFT and backprojection
- Few parameters
- Typically works very well
- Reconstruction behaviour well understood

# Filtered Back Projection (FBP)

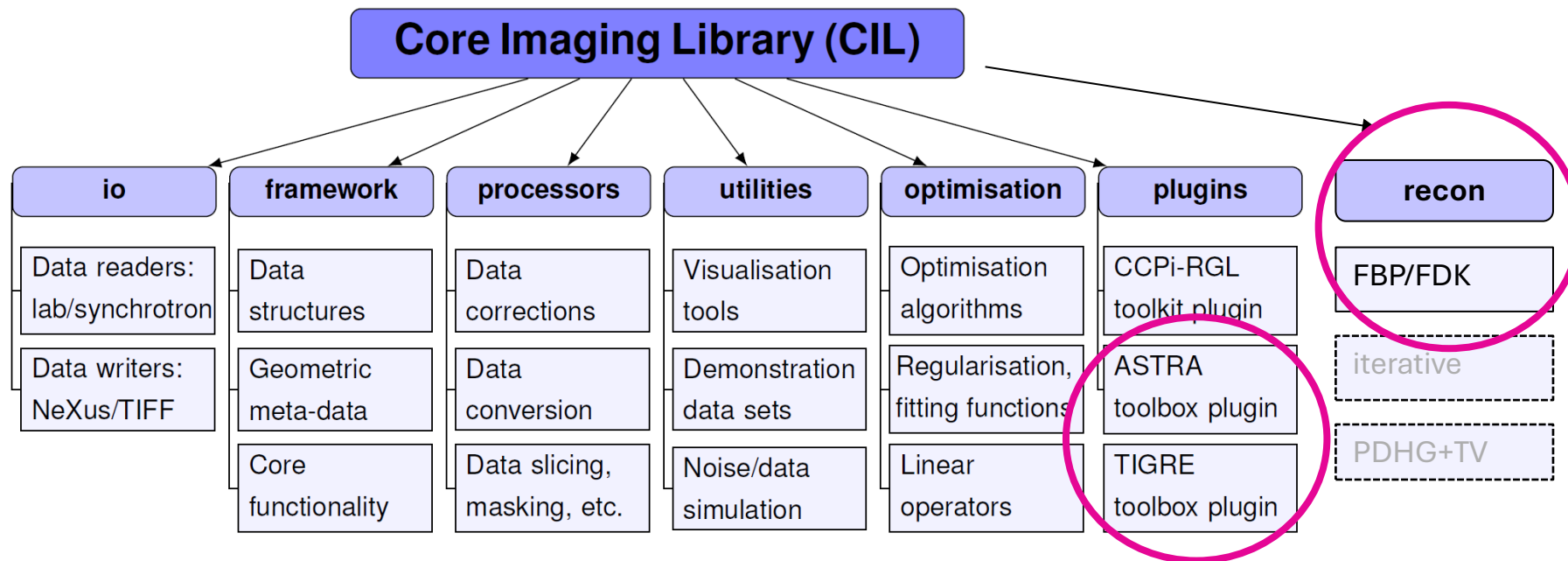
## Pros

- Fast as based on FFT and backprojection
- Few parameters
- Typically works very well
- Reconstruction behaviour well understood

## Cons



- Number of projections needed proportional to acquisition panel size
- Full angular range required (**limited angle** problem)
- Modest amount of noise tolerated
- Fixed scan geometries
- Cannot make use of prior knowledge such as non-negativity

# Filtered Back Projection in CIL



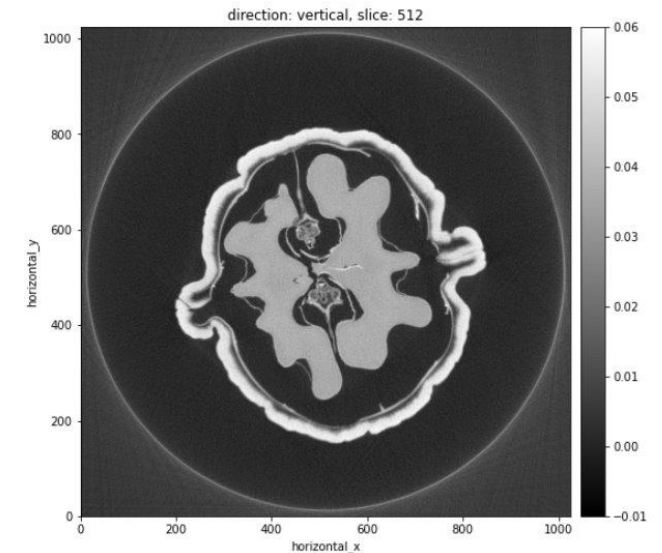
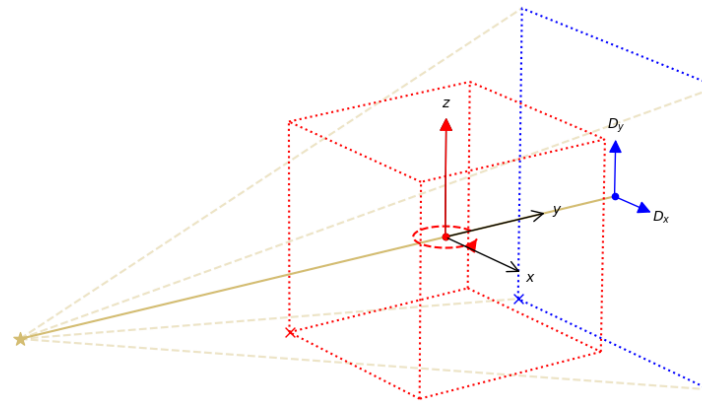


# Filtered Back Projection in CIL

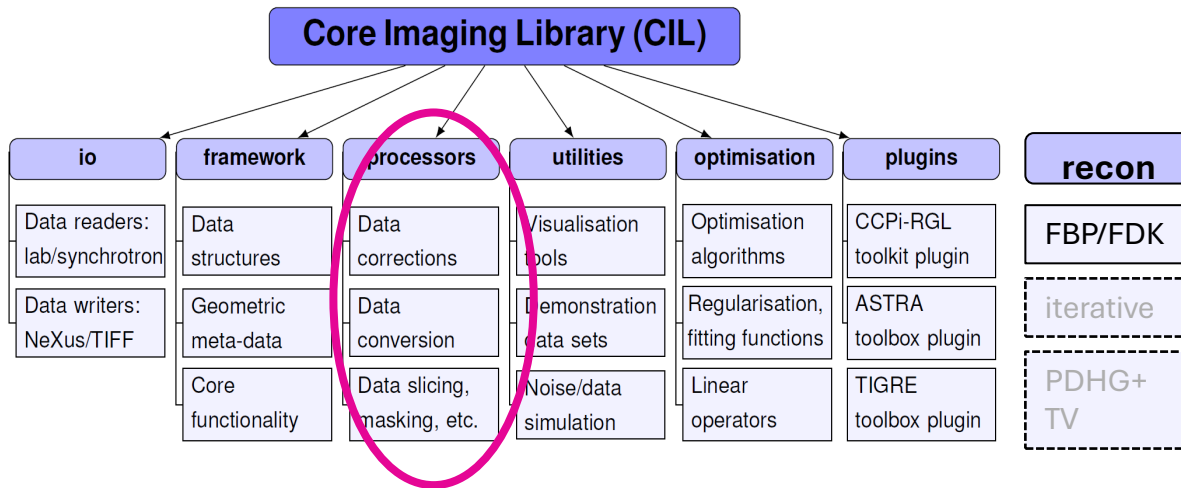
- Accelerated with TIGRE  or ASTRA  toolboxes
- Optimised standard algorithms for large data

```
data = ZEISSDataReader(filename).read()
data = TransmissionAbsorptionConverter()(data)
show_geometry(data.geometry)
recon = FDK(data).run()
show2D(recon)
```

— world coordinate system  
★ source position  
• rotation axis position  
— rotation axis direction  
... image geometry  
x data origin (voxel 0)  
• detector position  
— detector direction  
... detector  
x data origin (pixel 0)

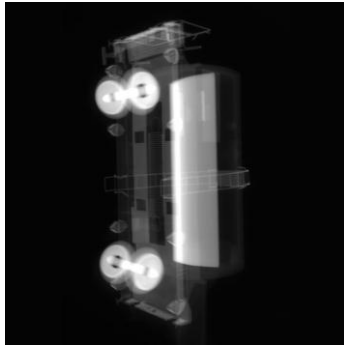
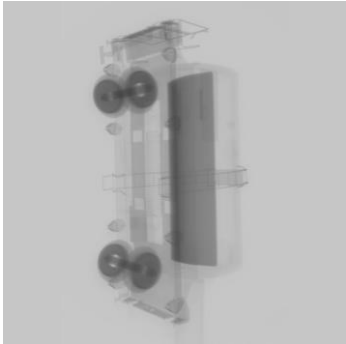


# Data Processing Methods

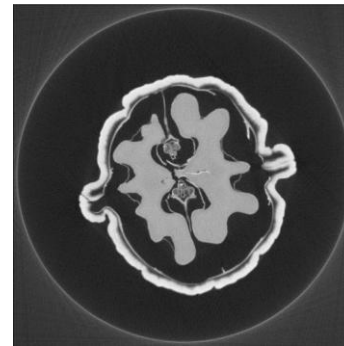
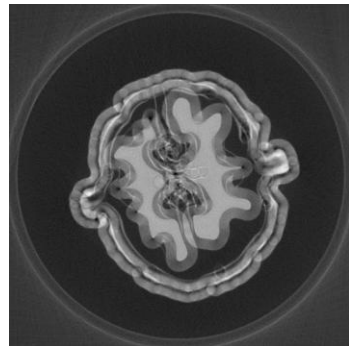


# Data processing methods

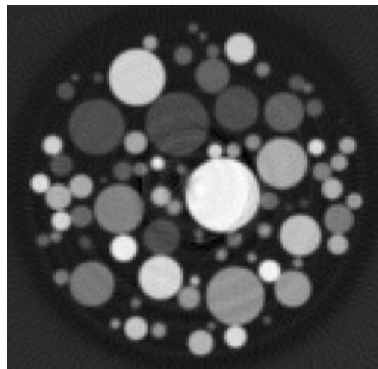
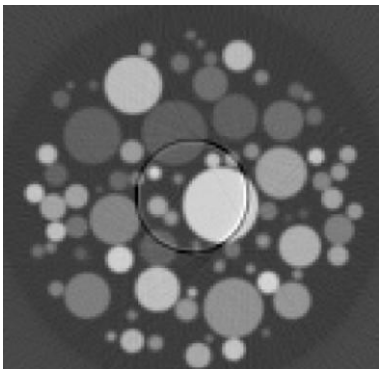
## Convert to absorption



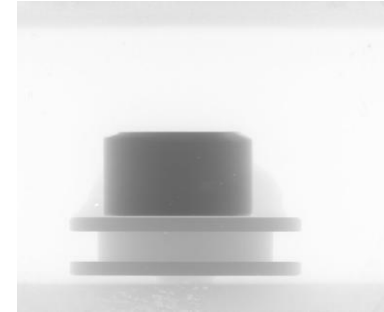
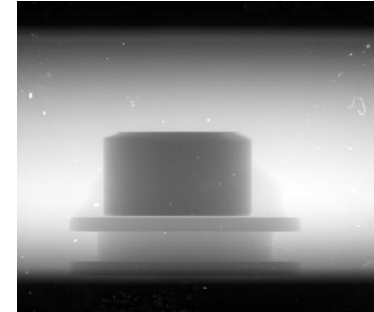
## Centre of Rotation correction



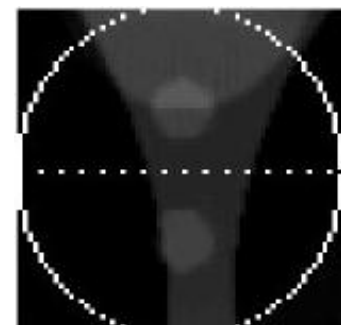
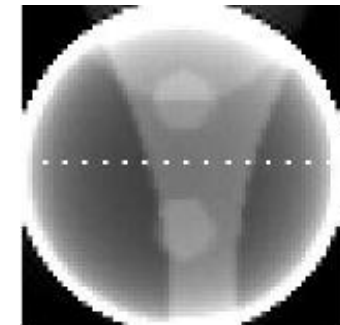
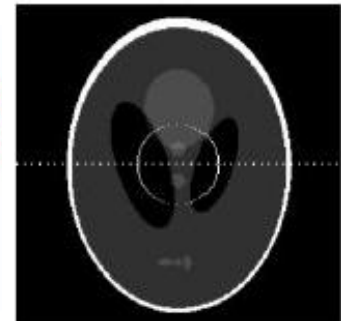
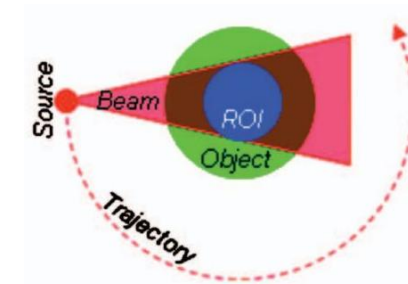
## Ring removal



## Flat-field correction



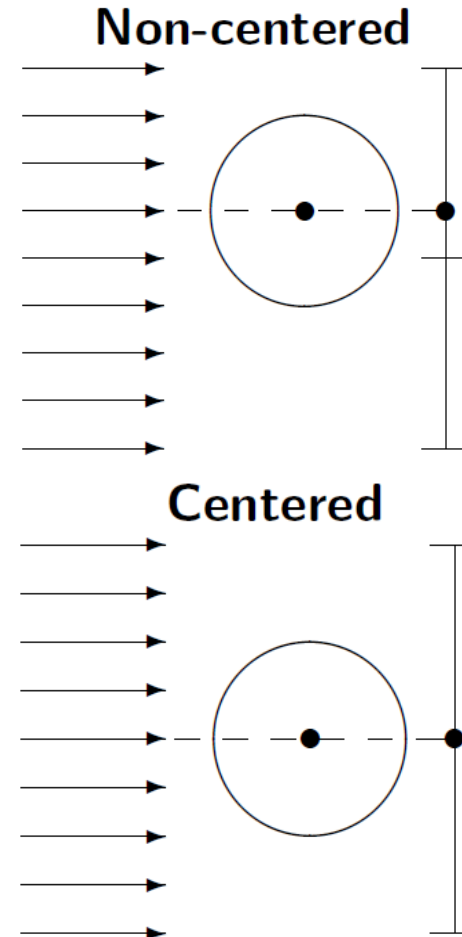
## Region of interest scans



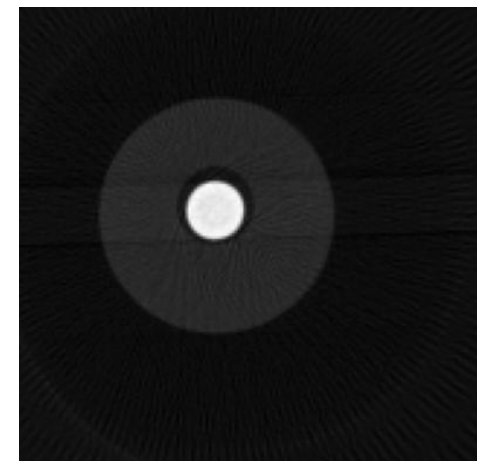
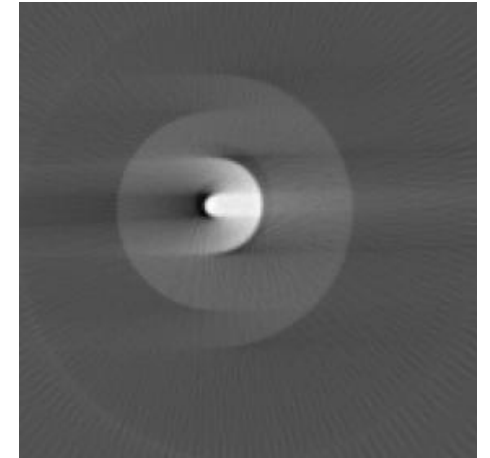
# Centre of Rotation Correction

- CoR: The projection of the axis of rotation onto the detector
- Reconstruction assumes axis of rotation is horizontally centred on detector
- Offset introduces blurring and artifacts
- We correct for this by updating our geometry

```
CentreOfRotationCorrector.image_sharpness()(data)
```



**Parallel Beam**

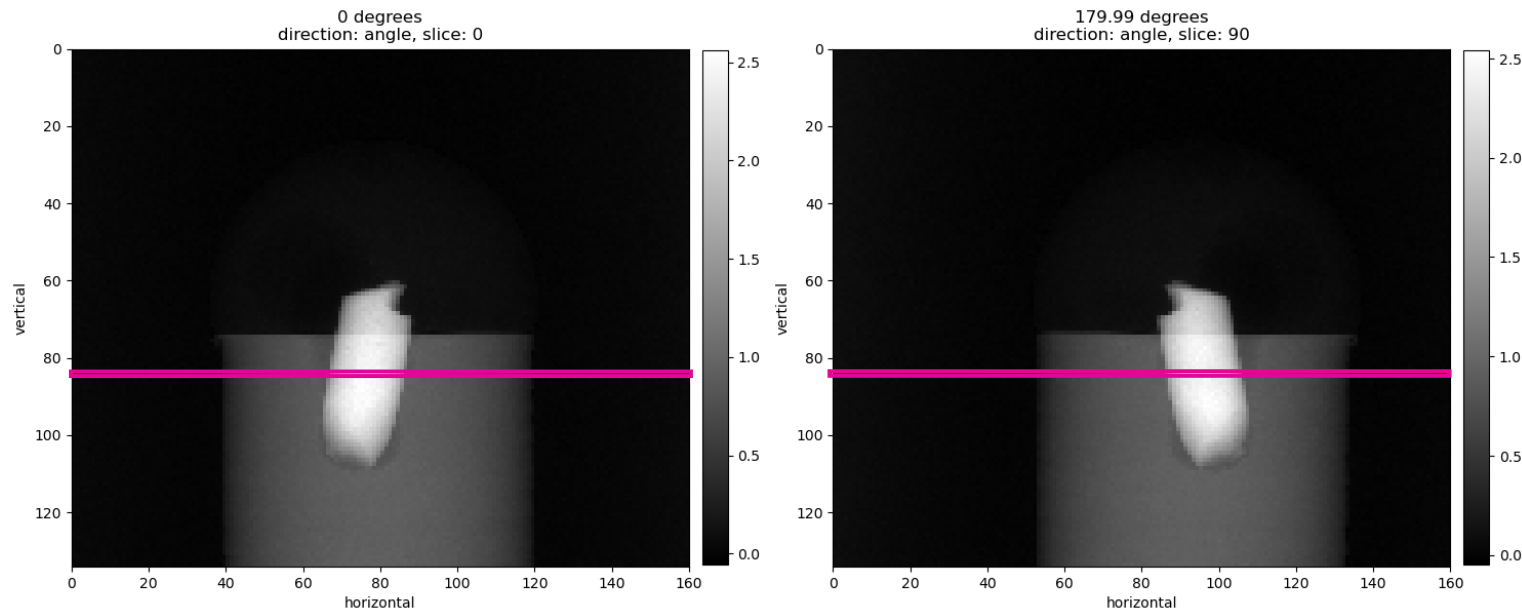




# Centre of Rotation in CIL – X Correlation

```
cil.processors.CentreOfRotationCorrector.xcorrelation(slice_index='centre',  
projection_index=0, ang_tol=0.1)
```

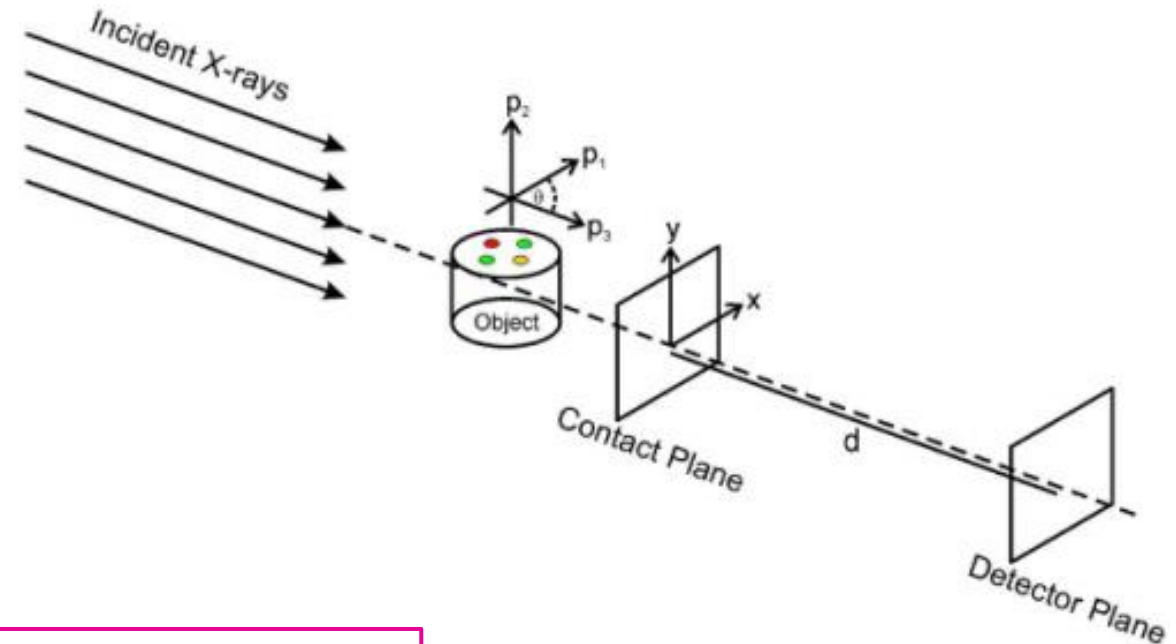
- For **parallel beam** geometry
- Requires two rows 180 degrees apart



# Phase retrieval

- Phase contrast imaging combines absorption data with phase information
- Can provide additional contrast for samples with similar materials
- Phase retrieval methods (e.g. Paganin) separate phase information and boost signal to noise (SNR)

```
processor = PaganinProcessor(delta=delta, beta=beta,  
                             energy=energy, energy_units='keV',  
                             full_retrieval=False)  
  
processor.set_input(data_before)  
data = processor.get_output()
```



# Log in to JupyterHub

- Exercises at CIL Jupyter notebook server:

<https://dev.platform.qim.dk/tools/jupyter-launcher>

## Jupyter Launcher

☒ Show detailed configuration

The screenshot shows the Jupyter Launcher configuration interface. The 'Project' dropdown is set to 'QIM platform'. The 'Root directory' is set to '/dtu/3d-imaging-center/courses/'. Below these, there are three configuration sections: 'CPUs' (set to 4), 'Memory (GB)' (set to 32), and 'Duration (Hours)' (set to 3). Each section has a slider below the input field. The 'CPUs', 'Memory (GB)', and 'Duration (Hours)' sections are highlighted with pink boxes.

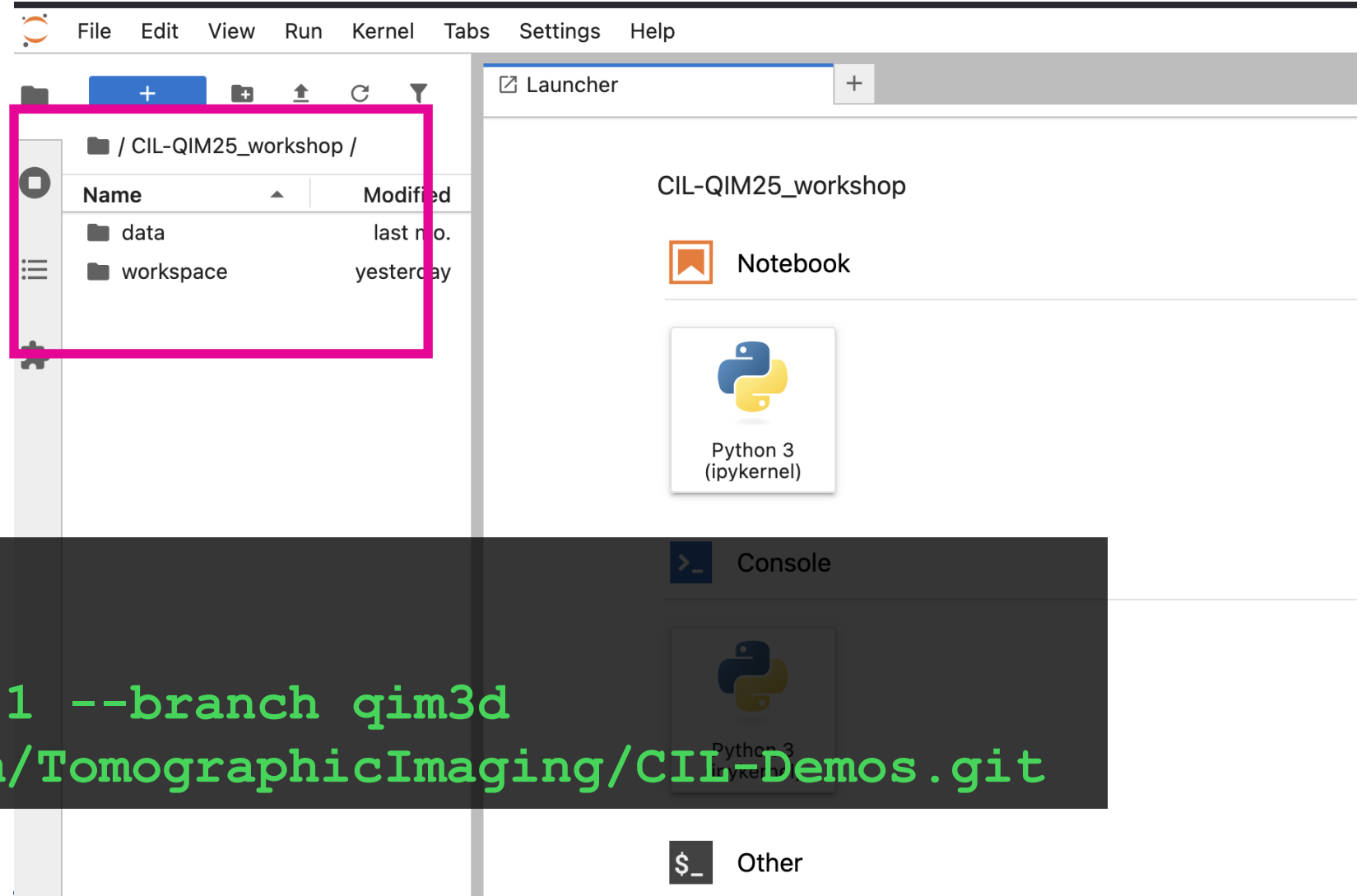
Configuration	Value
Project	QIM platform
Root directory	/dtu/3d-imaging-center/courses/
CPUs	4
Memory (GB)	32
Duration (Hours)	3

# Log in to JupyterHub

Hostname qimlogin.gbar.dtu.dk	<b>HPC queue</b> c47511	Job name Jupyter
<b>Conda environment</b> Pre-installed libraries for different tasks. Leave it empty if you want to use your own custom environment. cil_25-0 (Core Imaging Library, version 25.0)		
<div>Reset tunnels to the platform, may solve connection issues. <input checked="" type="checkbox"/> Reset SSH tunnels</div> <div><b>Jupyter server type</b> Select Lab in case you want to use extensions. <input type="radio"/> Notebook <input checked="" type="radio"/> Lab</div> <div>Show logs of the job submission. They will be visible after the process starts. <input type="checkbox"/> Show logs</div>		
Launch Jupyter Server		

# Log in to JupyterHub

Open terminal



The screenshot shows the JupyterHub web interface. The top navigation bar includes 'File', 'Edit', 'View', 'Run', 'Kernel', 'Tabs', 'Settings', and 'Help'. Below this, the 'Launcher' tab is active, displaying the 'CIL-QIM25\_workshop' directory. A pink rectangle highlights the file browser on the left, which shows a table of files and folders:

Name	Modified
data	last mo.
workspace	yesterday

Below the file browser, the 'Console' tab is visible, showing a terminal window with the following commands:

```
cd workspace
mkdir yourname
git clone --depth 1 --branch qim3d
https://github.com/TomographicImaging/CIL-Demos.git
```

# CIL ESRF pipeline

CIL-Demos/demos/4\_Deep\_Dives/05\_esrf\_pipeline.ipynb

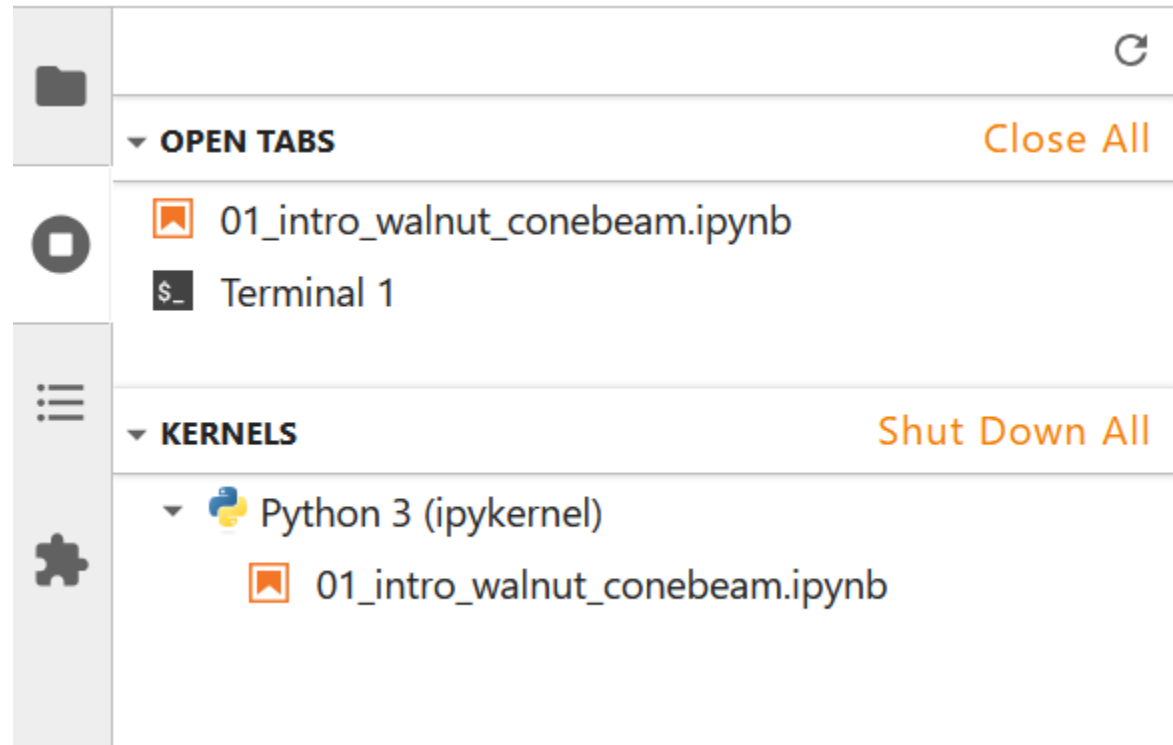
- Pipeline specifically designed for ESRF data
- Configurable for reading different datasets
- Demonstrating pre-processors for common artefacts in synchrotron datasets

**20 minutes to run the notebook, then discussion**

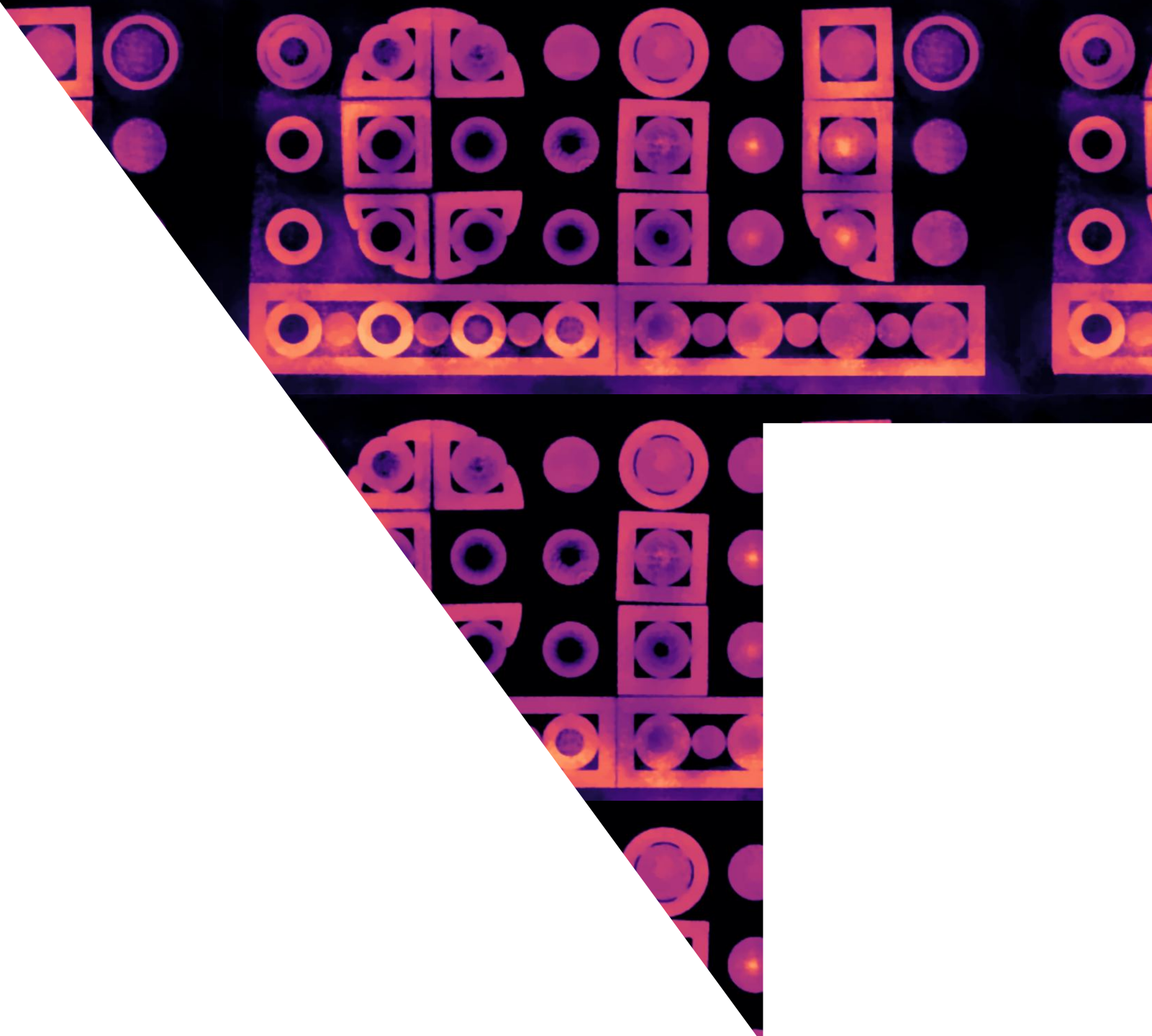


# CIL ESRF pipeline discussion

# Shut Down Notebooks



**Over to you!**



# Exercises

- Notebooks with lines for you to fill in!
- Either:
  - More investigation of pre-processing methods:

[CIL-Demos/demos/1\\_Introduction/exercises/02\\_preprocessing\\_seeds\\_conebeam.ipynb](#)

- Try configuring a custom data reader:

[CIL-Demos/demos/1\\_Introduction/exercises/03\\_where\\_is\\_my\\_reader.ipynb](#)

**25 minutes to run the notebook, then discussion**

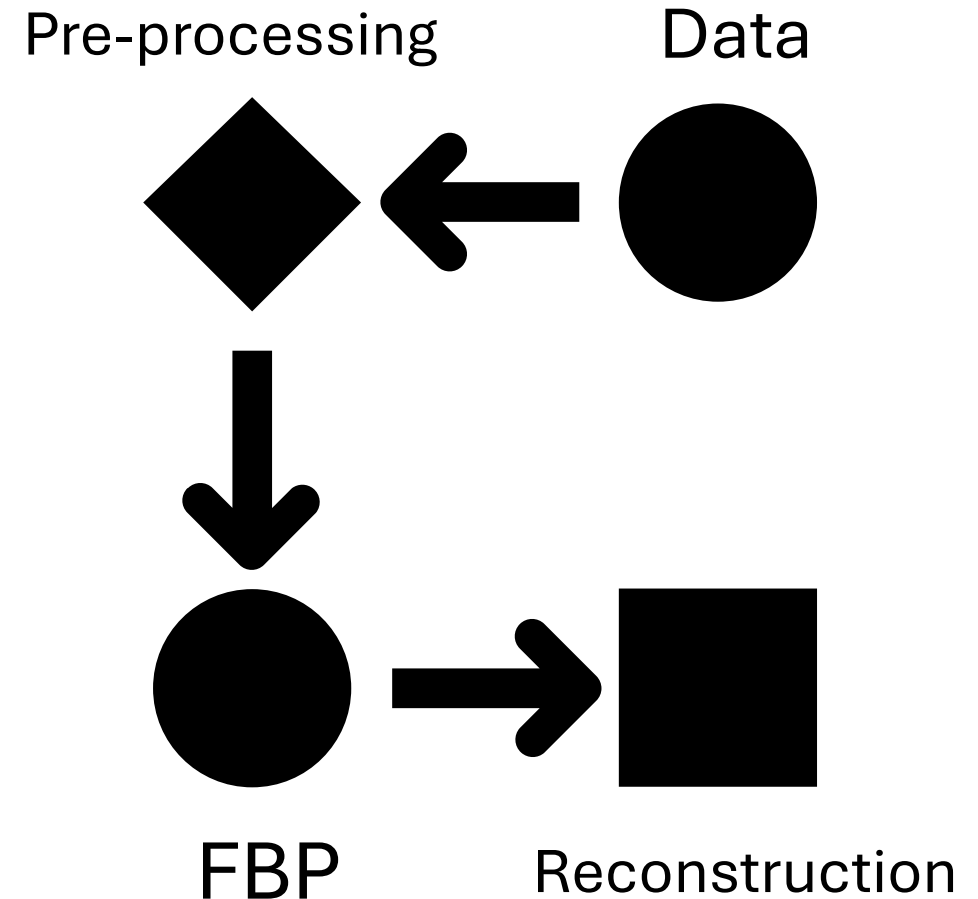
# Exercise discussion

# Wrap up

Filtered back-projection is **very good!**

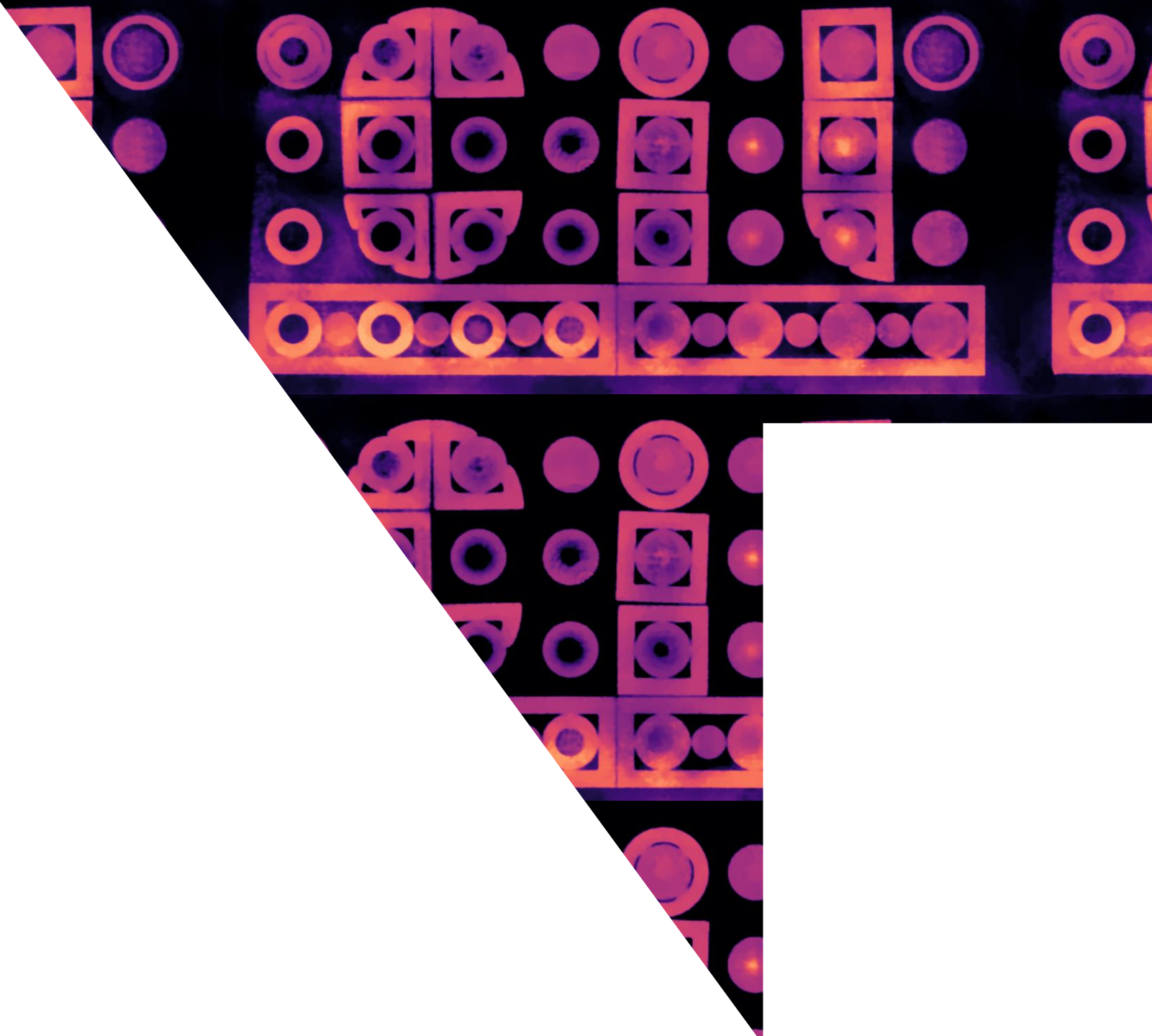
- If your data is good it should work well
- Make sure you've defined your geometry correctly
- Do any necessary pre-processing

If your data is not good... consider other methods

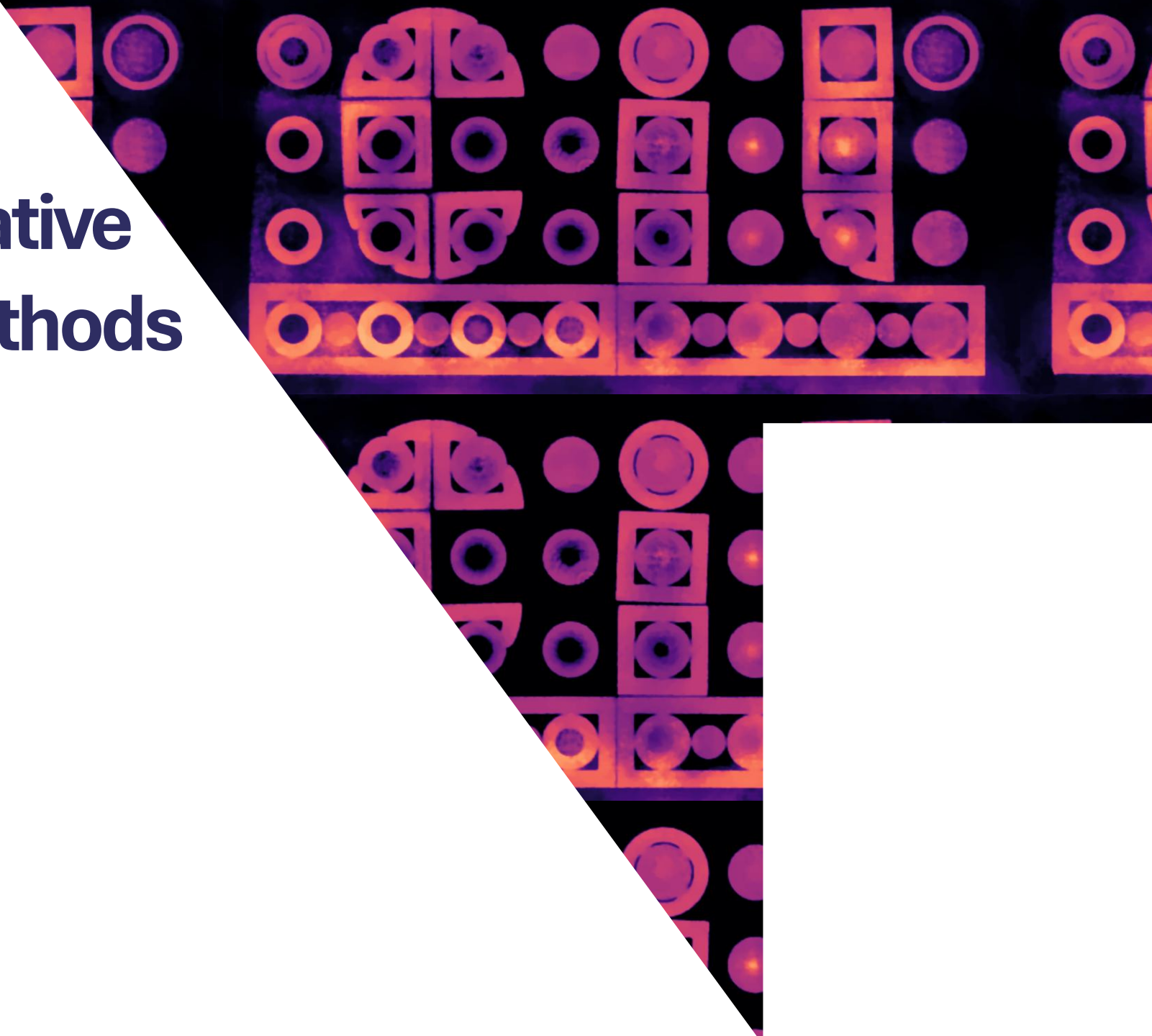




**30min break**



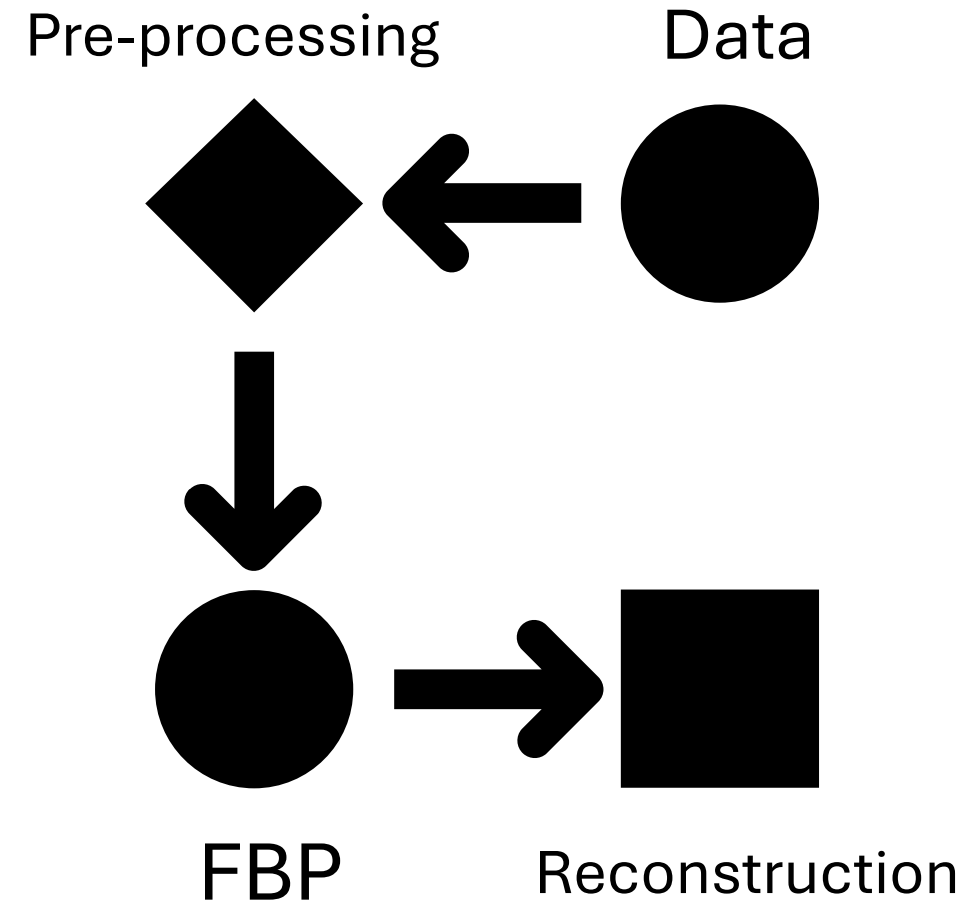
# Introduction to Iterative Reconstruction Methods



# Filtered Back Projection (FBP)

## Pros

- Fast as based on FFT and backprojection
- Few parameters
- Typically works very well
- Reconstruction behaviour well understood



# Filtered Back Projection (FBP)

## Pros

- Fast as based on FFT and backprojection
- Few parameters
- Typically works very well
- Reconstruction behaviour well understood

## Cons

- Number of projections needed proportional to acquisition panel size
- Full angular range required (**limited angle** problem)
- Modest amount of noise tolerated
- Fixed scan geometries
- Cannot make use of prior knowledge such as non-negativity

# Take-away

Filtered back-projection is **very good!**

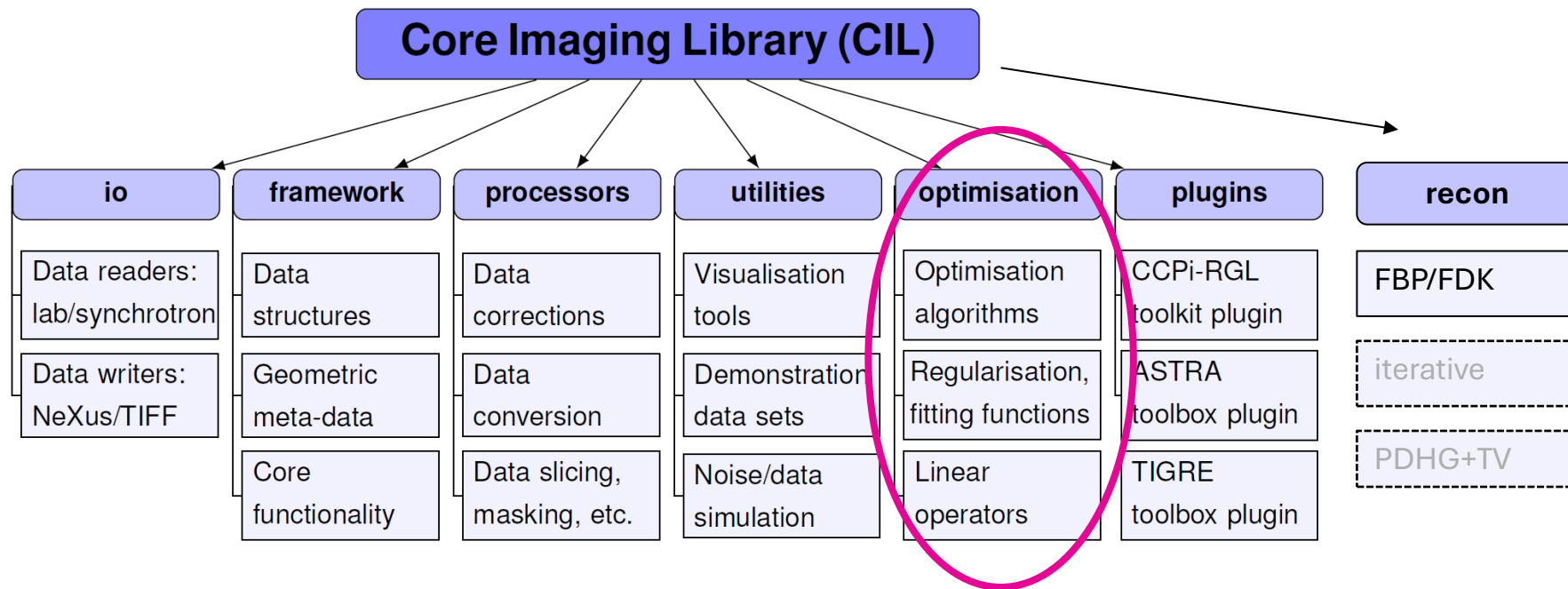
If data is good, look no further!

If data is **bad**, iterative reconstruction may help, **but**

Different kinds of **bad** need different methods.

CIL provides a range of **iterative reconstruction methods** for CT and other inverse problems

# Iterative reconstruction methods





# Imaging Model for Iterative Reconstruction

$$\frac{I}{I_0} = \exp \int_{L_i} -\mu(s) ds$$

$$b_i = -\log \frac{I_i}{I_0} = \int_{L_i} \mu(s) ds$$

$$b_i = \sum_j a_{ij} u_j \rightarrow Au = b$$

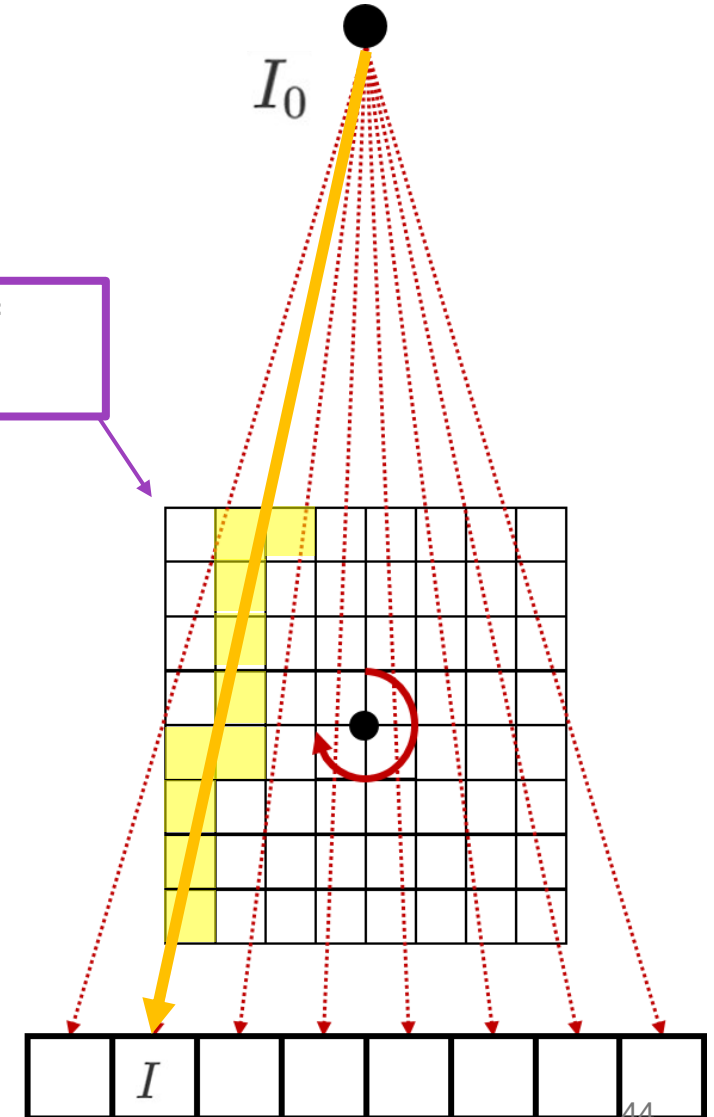
- Assume the object is constant in each pixel
- $u_j$  is the  $j$ -th pixel value
- $a_{ij}$  is the path length in the  $j$ -th pixel

X-ray source

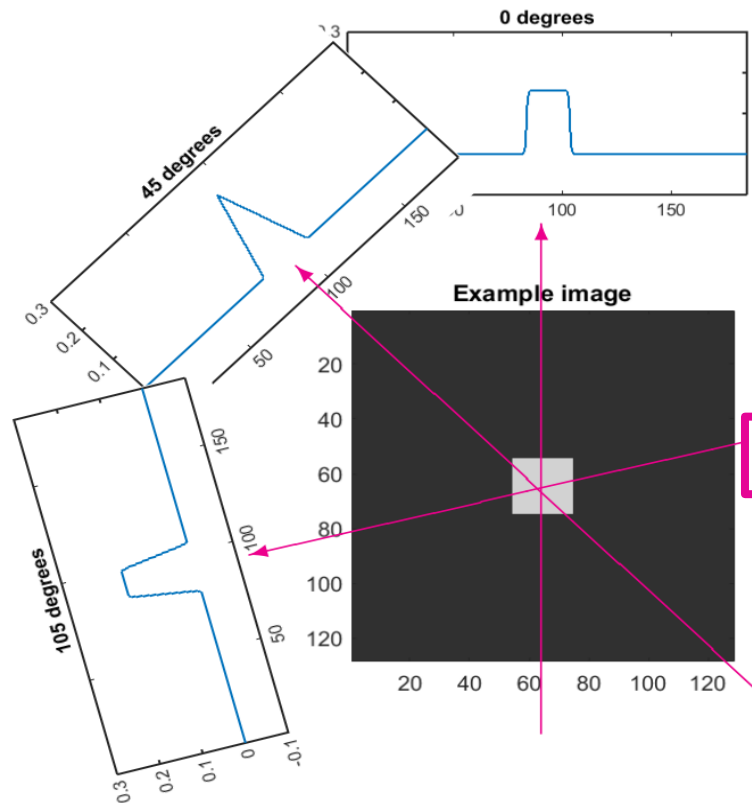
$I_0$

Extremely large set of linear equations!

Measurement volume



# Iterative Reconstruction



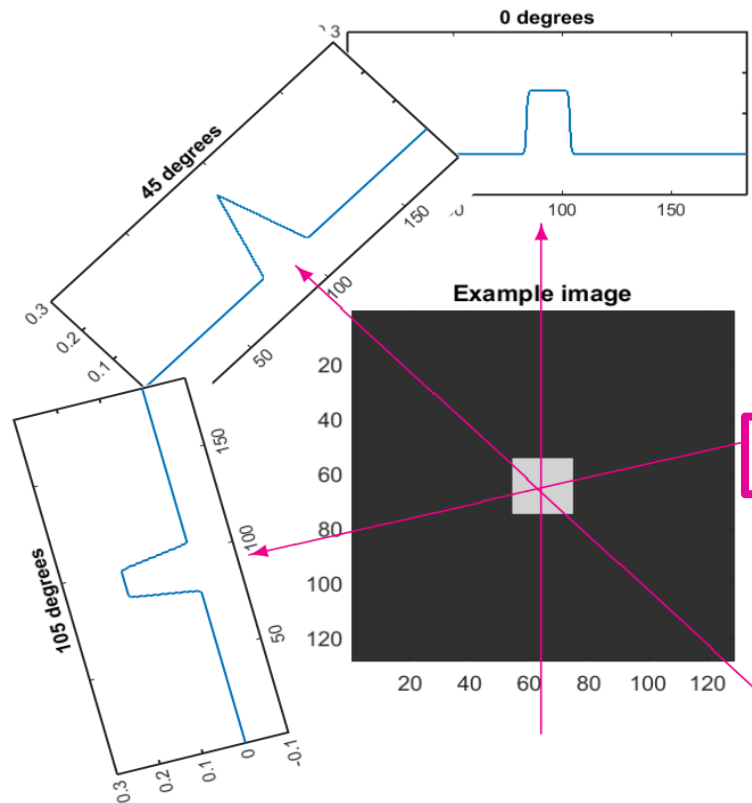
**b**: measured data (projections)

**A**: physics model =  
forward projector

**u**: reconstructed volume

$$Au = b$$

# Iterative Reconstruction



**b**: measured data (projections)

**A**: physics model =  
forward projector

$$Au = b$$

**u**: reconstructed volume

Which we solve iteratively

$$u^{\star} = \underset{u}{\operatorname{argmin}} \{ \mathcal{D}(Au, b) \}$$

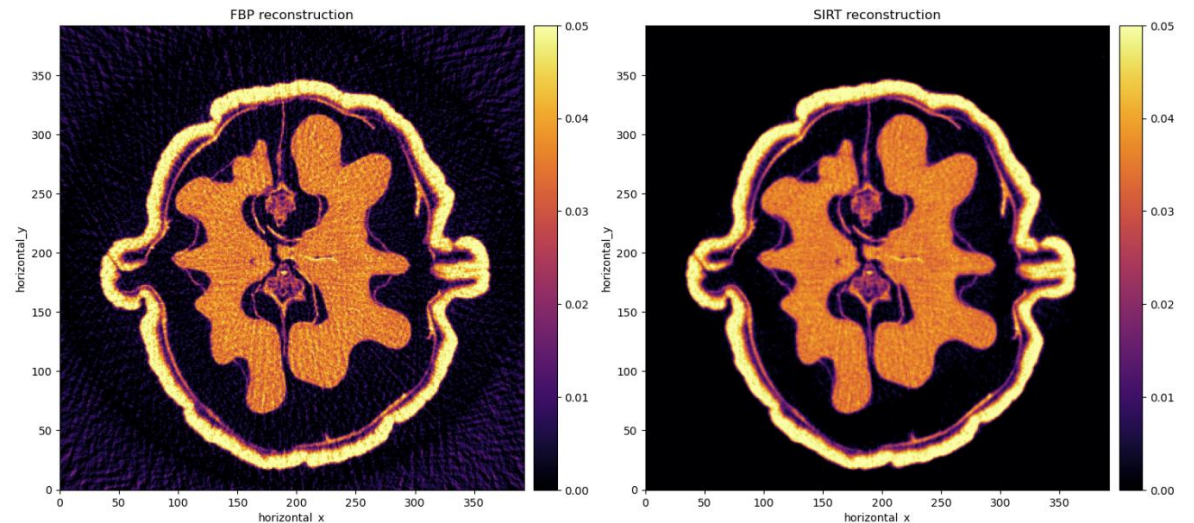
# Iterative Reconstruction in CIL

$$Au = b$$

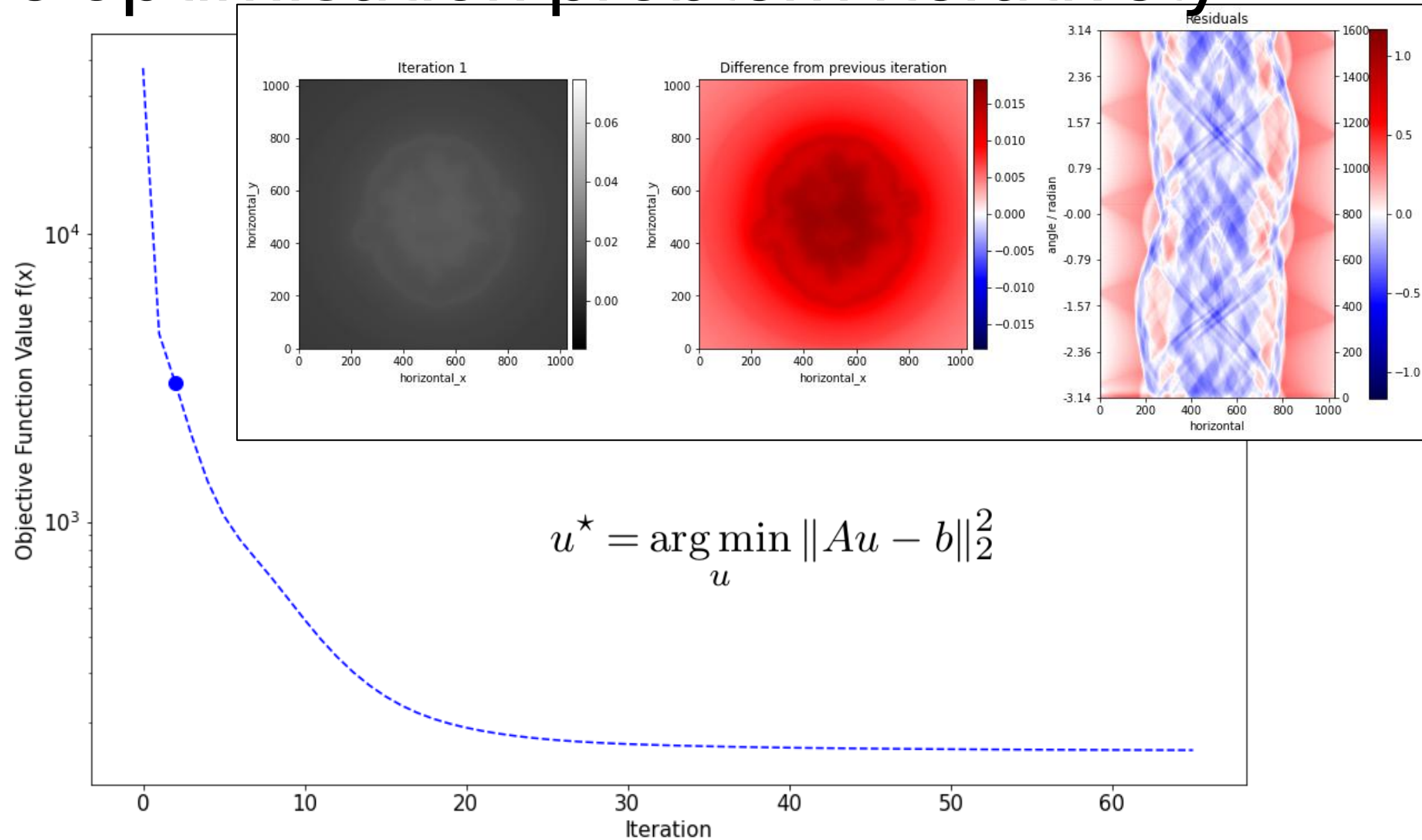
Construct the iterative reconstruction method based on **optimisation algorithms** and **objective functions**

$$u^* = \underset{u}{\operatorname{argmin}} \{ \mathcal{D}(Au, b) \}$$

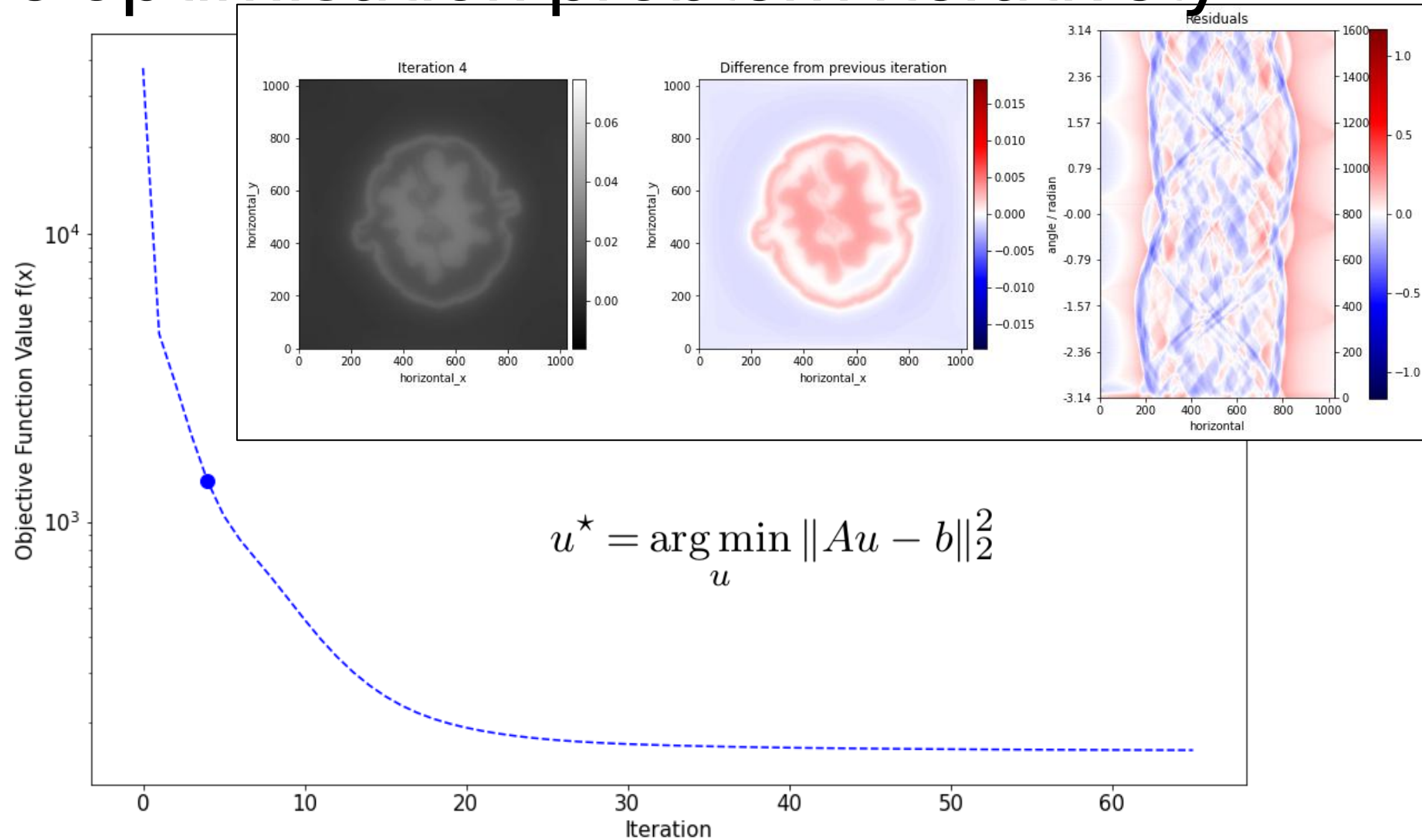
```
A = ProjectionOperator(ig, ag)
init = ig2D.allocate(0)
sirt = SIRT(initial = x_init, operator = A , data = b)
sirt.run(300, verbose=1)
sirt_recon = sirt.solution
show2D([fbp_recon,sirt_recon],
       title = ['FBP reconstruction','SIRT reconstruction'],
       cmap = 'inferno', fix_range=(0,0.05))
```



# Solve optimisation problem iteratively

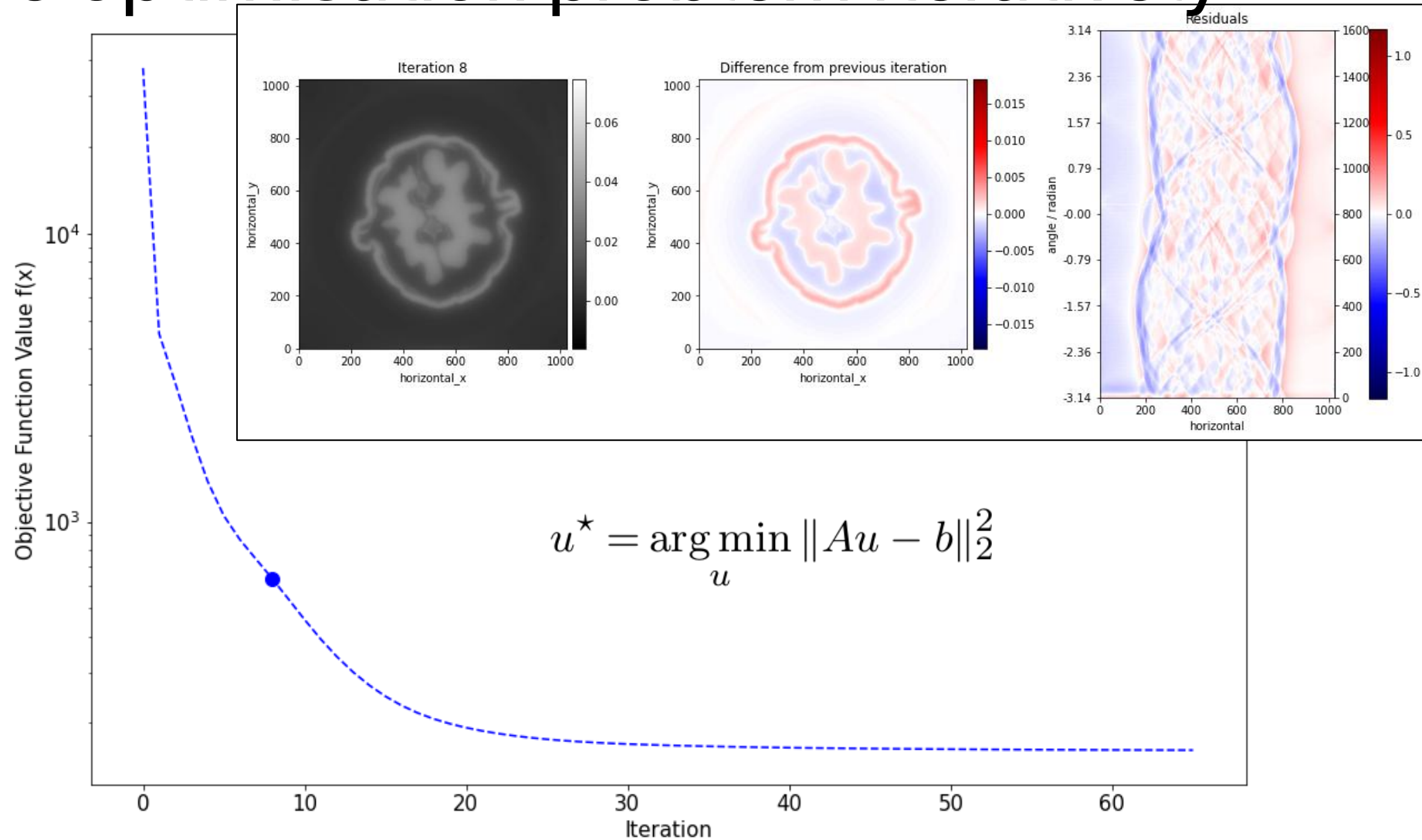


# Solve optimisation problem iteratively

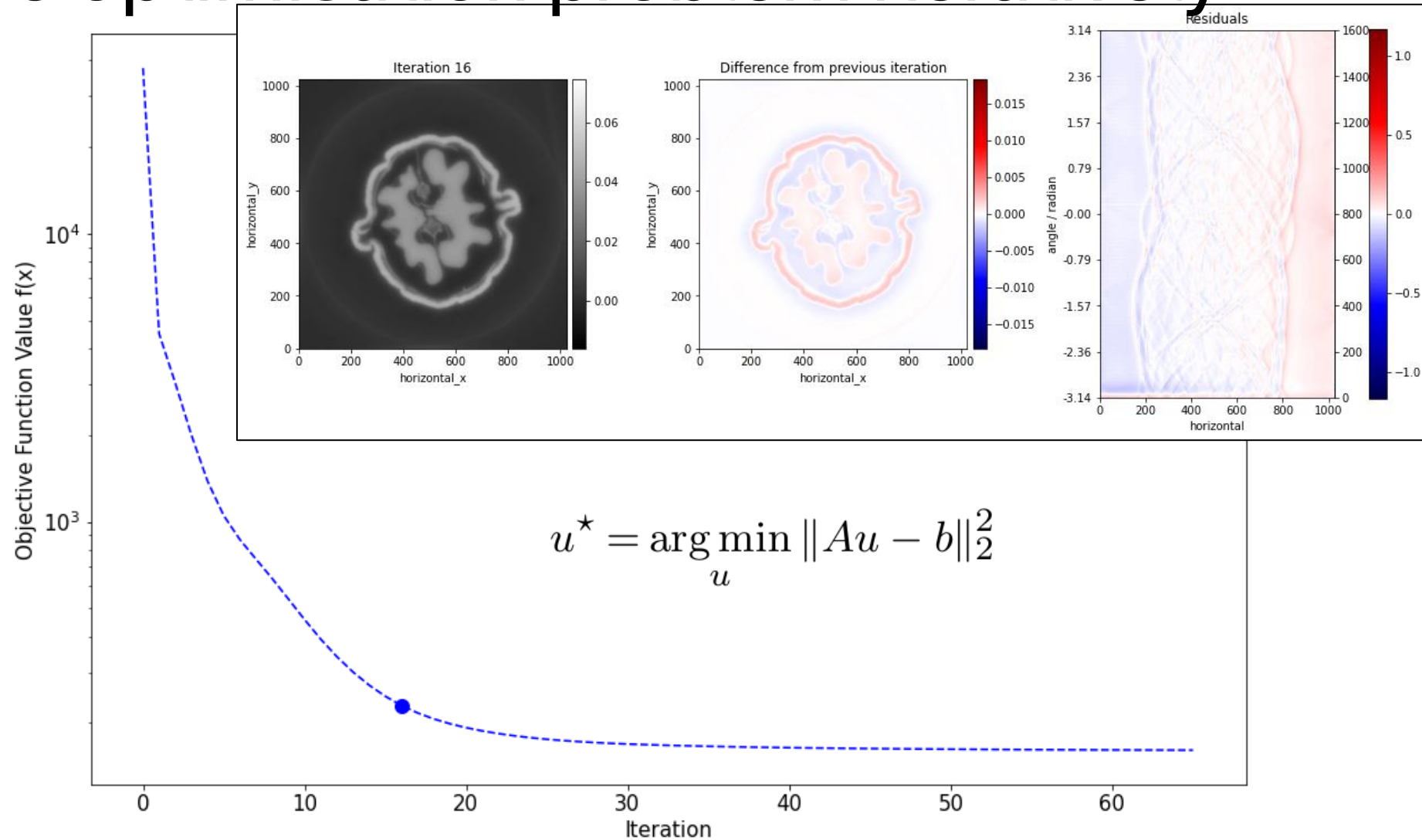




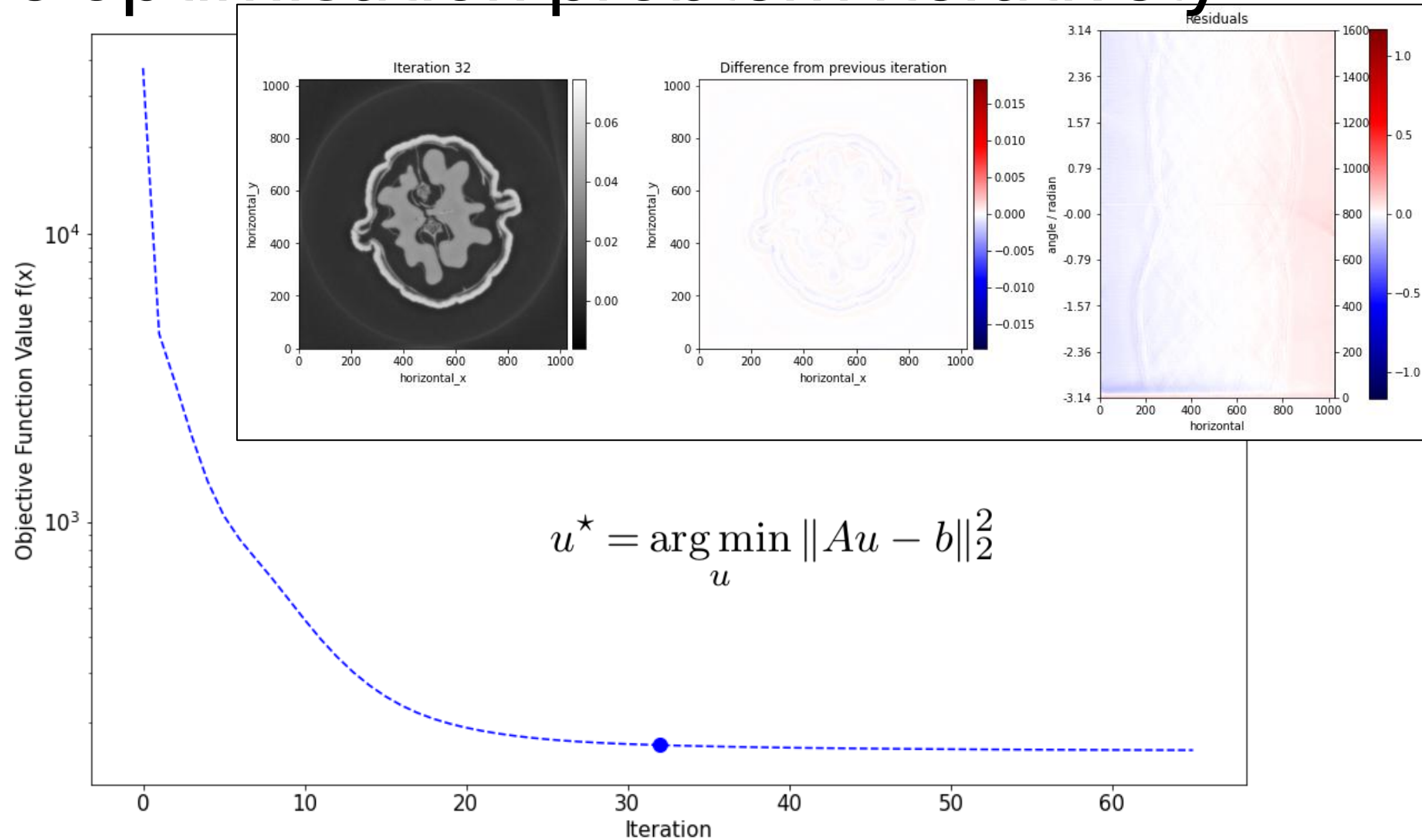
# Solve optimisation problem iteratively



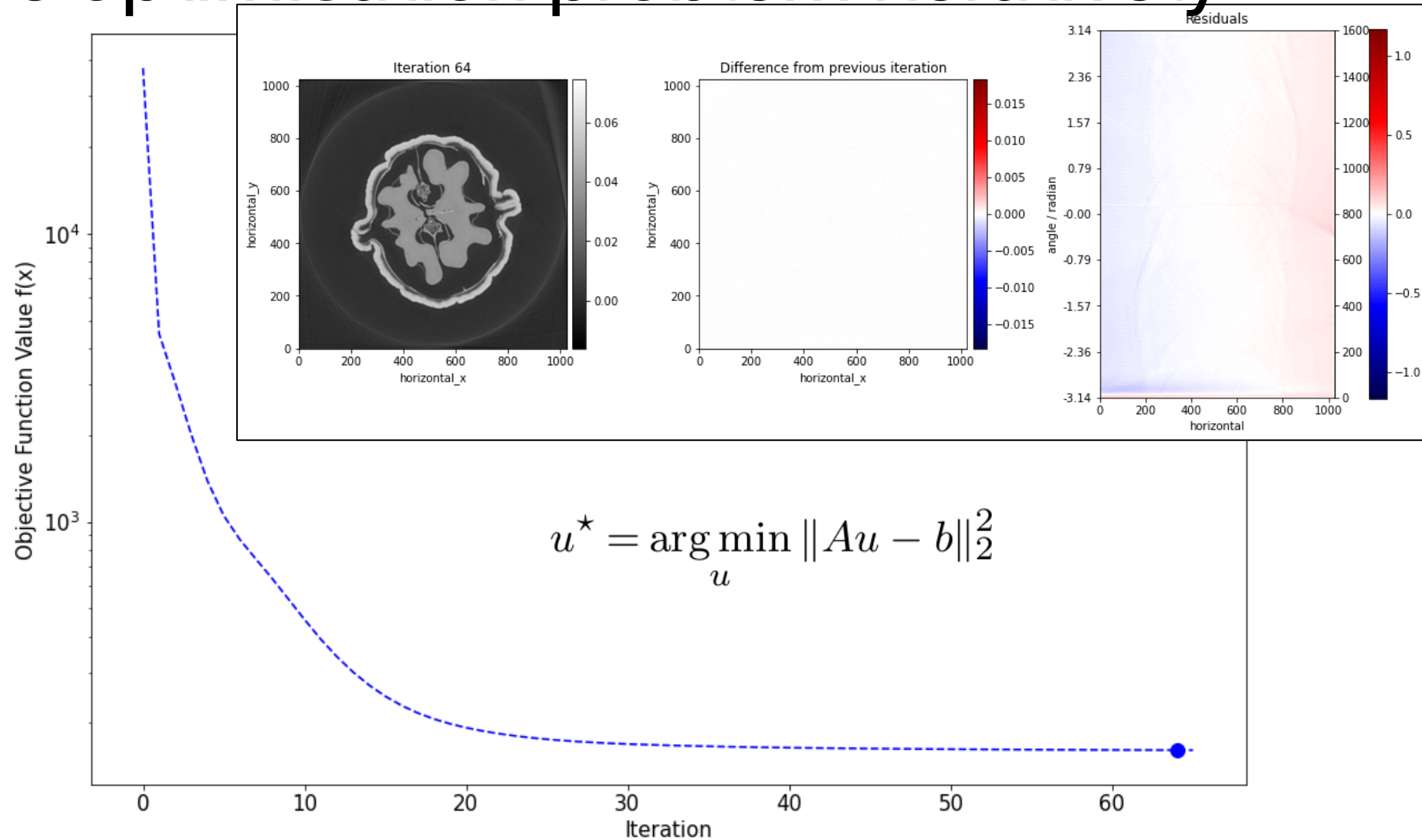
# Solve optimisation problem iteratively



# Solve optimisation problem iteratively



# Solve optimisation problem iteratively



# Iterative reconstruction demo

CIL-Demos/demos/1\_Introduction/05\_usb\_limited\_angle\_fbp\_sirt  
notebook

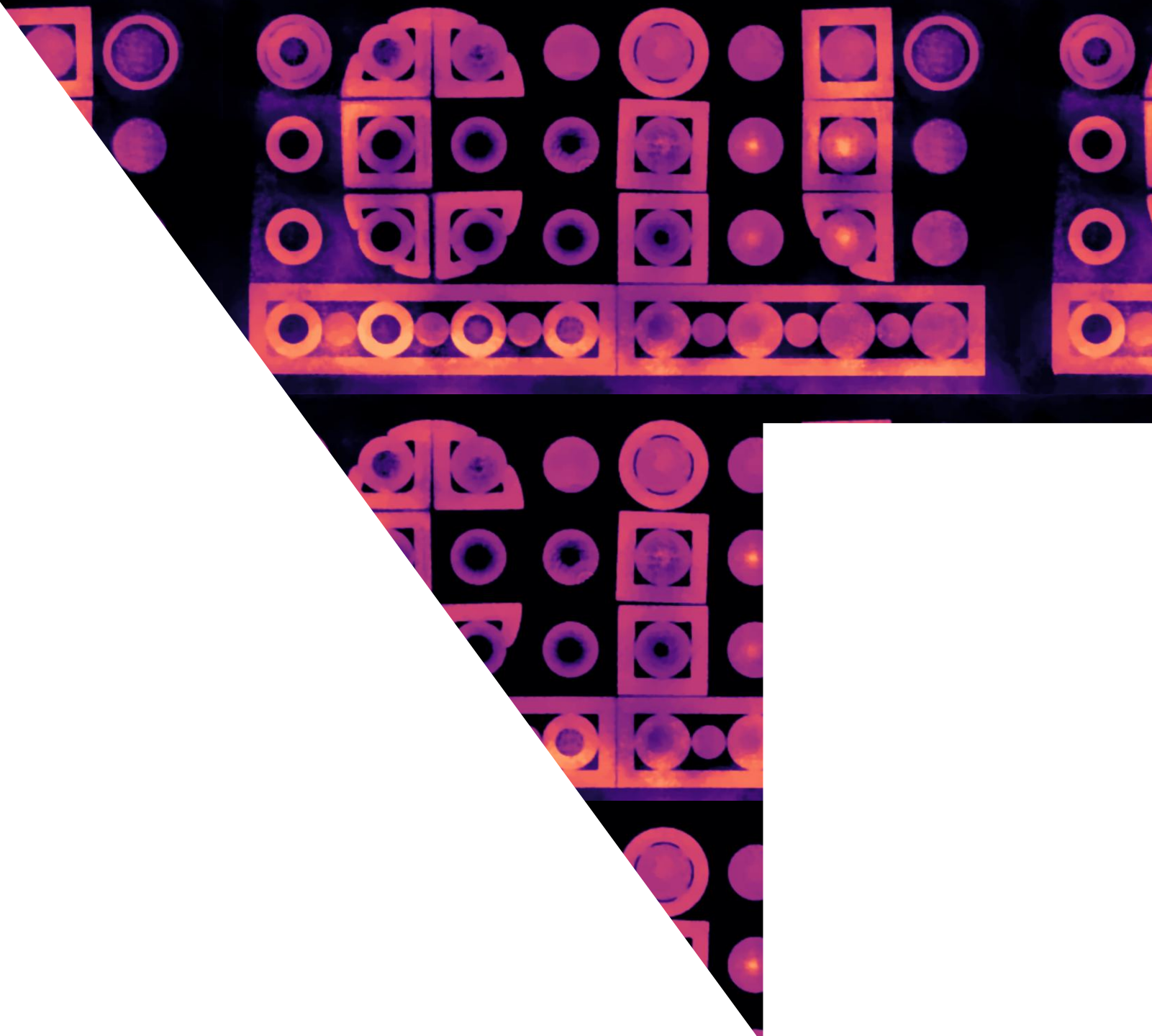
- Compare SIRT reconstruction to FBP reconstruction on limited angle data.
- Modify image geometry to reduce reconstruction volume to save memory and time.

**20 minutes to run the notebook, then discussion**

Extension: run with different numbers of angles or try  
1\_Introduction/04\_FBP\_CGLS\_SIRT.ipynb

# Iterative reconstruction demo - discussion

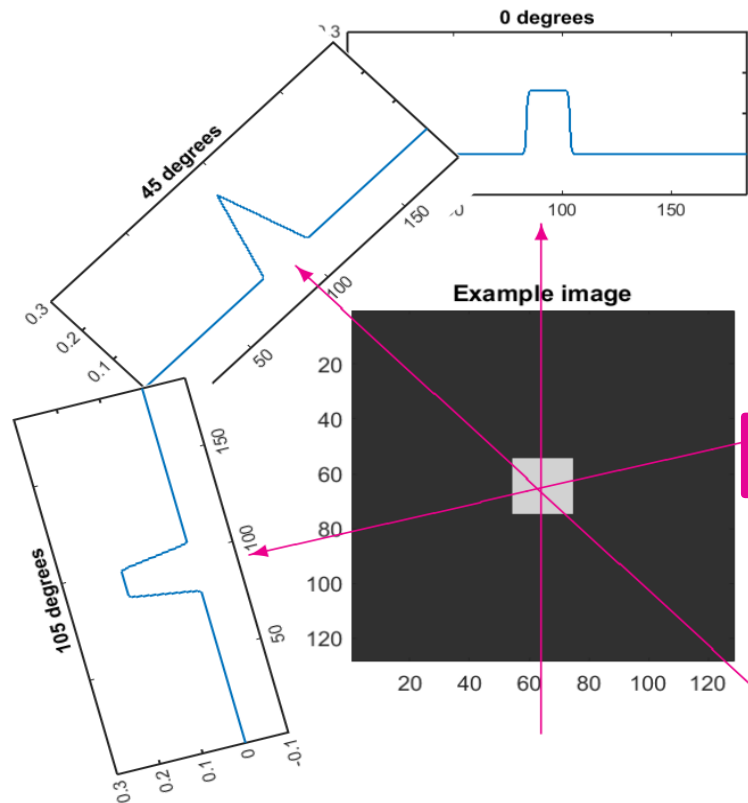
# Introduction to Regularisation





# Iterative Reconstruction

Ill posed problem



**b**: measured data (projections)

**A**: physics model =  
forward projector

**u**: reconstructed volume

$$Au = b$$

In case either:

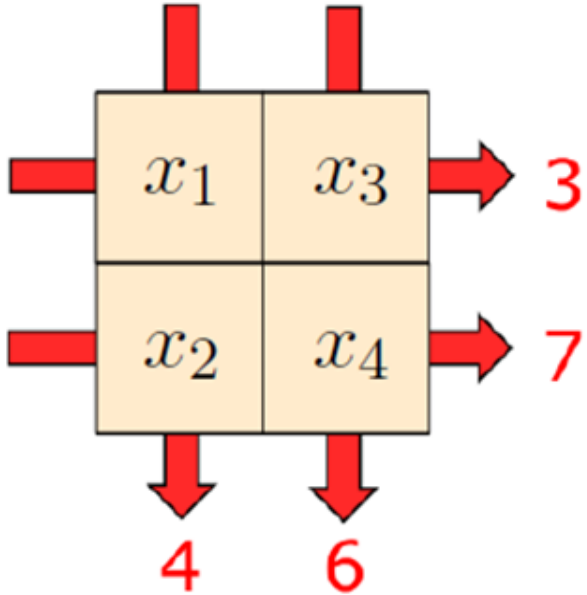
1. No solution
2. Not unique solution
3. Solution sensitive to noise

0. Modelling errors in **A**

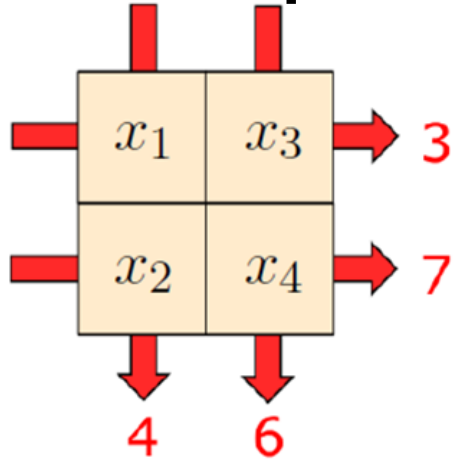
Which we solve iteratively

$$u^{\star} = \underset{u}{\operatorname{argmin}} \{ \mathcal{D}(Au, b) + \alpha \cdot \mathcal{R}(u) \}$$

# Simple analogy

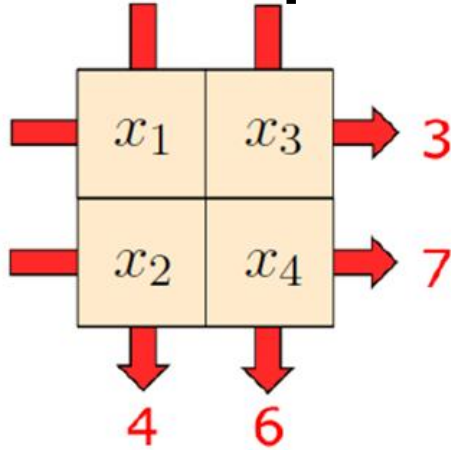


# Simple analogy



$$\begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} 3 \\ 7 \\ 4 \\ 6 \end{pmatrix}$$

# Simple analogy

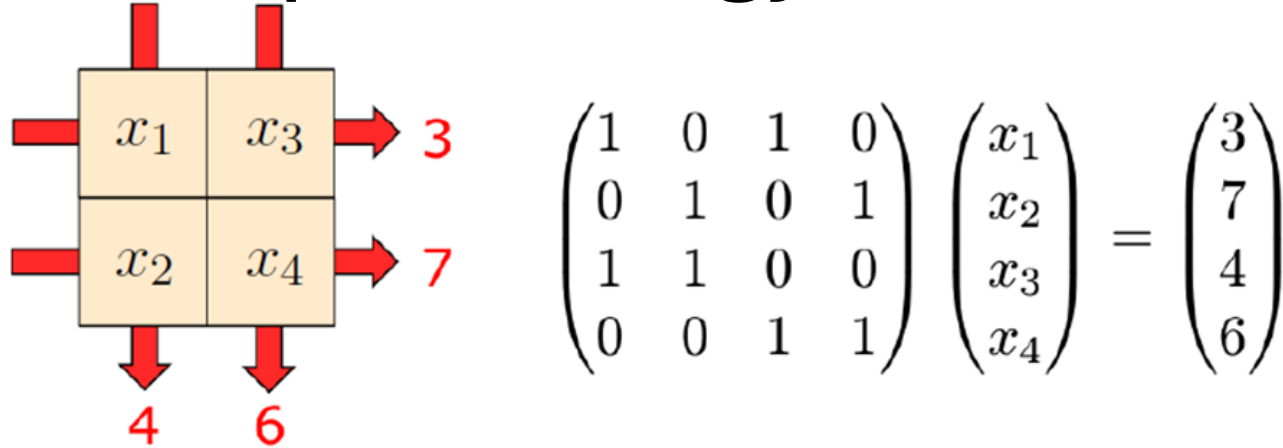


$$\begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} 3 \\ 7 \\ 4 \\ 6 \end{pmatrix}$$

Infinitely many solutions ( $k \in \mathbb{R}$ ):

$$\begin{pmatrix} x_1 & x_3 \\ x_2 & x_4 \end{pmatrix} = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix} + k \times \begin{pmatrix} -1 & 1 \\ 1 & -1 \end{pmatrix}$$

# Simple analogy



Infinitely many solutions ( $k \in \mathbb{R}$ ):

$$\begin{pmatrix} x_1 & x_3 \\ x_2 & x_4 \end{pmatrix} = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix} + k \times \begin{pmatrix} -1 & 1 \\ 1 & -1 \end{pmatrix}$$

**Prior:** solution is integer and non-negative



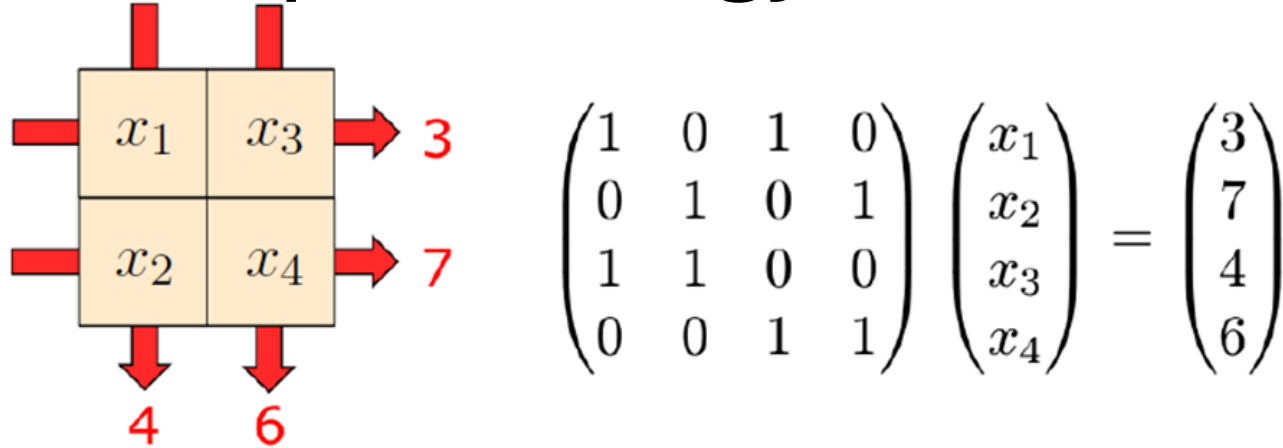
0	3
4	3

1	2
3	4

2	1
2	5

3	0
1	6

# Simple analogy



Infinitely many solutions ( $k \in \mathbb{R}$ ):

$$\begin{pmatrix} x_1 & x_3 \\ x_2 & x_4 \end{pmatrix} = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix} + k \times \begin{pmatrix} -1 & 1 \\ 1 & -1 \end{pmatrix}$$

**Prior:** solution is integer and non-negative



0	3
4	3

1	2
3	4

Sum of squares of the values is smallest

2	1
2	5

3	0
1	6

# Iterative Reconstruction in CIL

$$Au = b$$

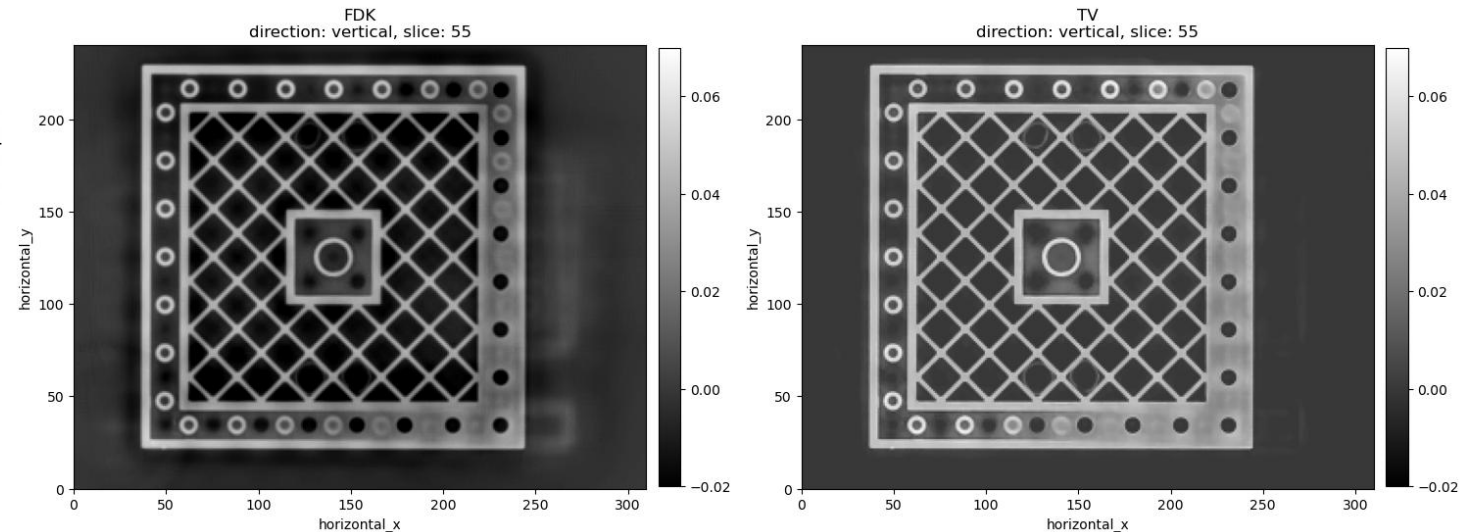
Construct the iterative reconstruction method based on **optimisation algorithms** and **objective functions**

$$u^* = \underset{u}{\operatorname{argmin}} \{ \mathcal{D}(Au, b) + \alpha \cdot \mathcal{R}(u) \}$$

```
Projector = ProjectionOperator(ig, ag)
LS = LeastSquares(A=Projector, b=data)
TV = FGP_TV(alpha=0.05, nonnegativity=True, device='gpu')

fista_TV = FISTA(initial=FDK_reco, f=LS, g=TV,
                  max_iteration=1000, update_objective_interval=10)
fista_TV.run(100)
TV_reco = fista_TV.solution

show2D([FDK_recon, TV_recon])
```

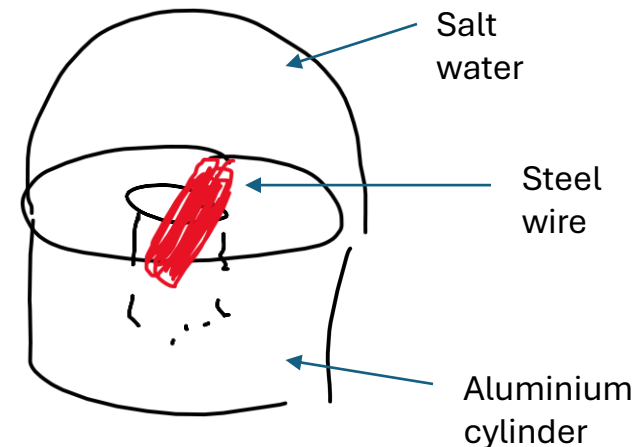




# Demonstration dataset

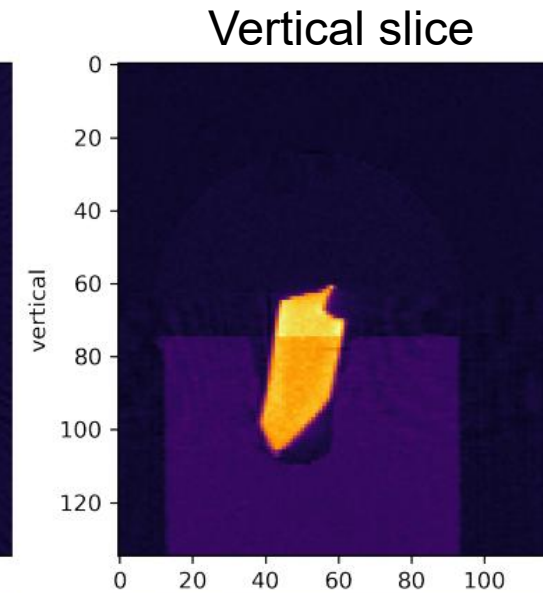
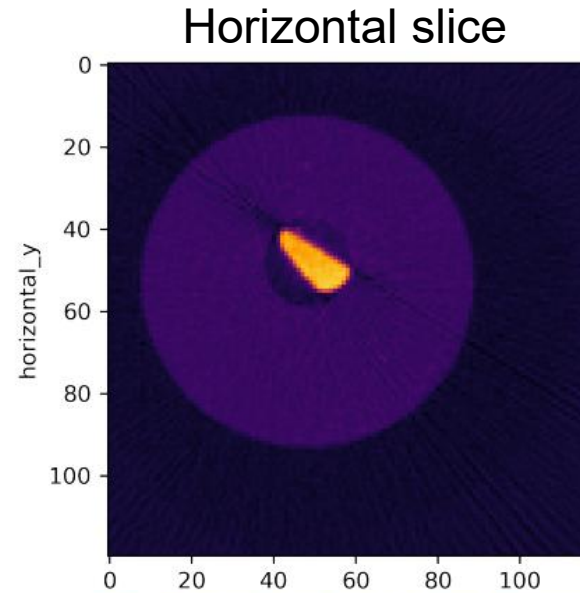
- 3D parallel-beam X-ray CT dataset from Beamline I13-2, Diamond Light Source.
- 0.5 mm aluminium cylinder with a piece of steel wire embedded in a small drilled hole. A droplet of salt water was placed on top, causing corrosion to form hydrogen bubbles.
- 160x135 15 projections over 180°

Jørgensen et al.: *Core Imaging Library - Part I: a versatile Python framework for tomographic imaging* Phil. Trans. R. Soc. A. **379** 20200192 (2021) DOI: [10.1098/rsta.2020.0192](https://doi.org/10.1098/rsta.2020.0192)

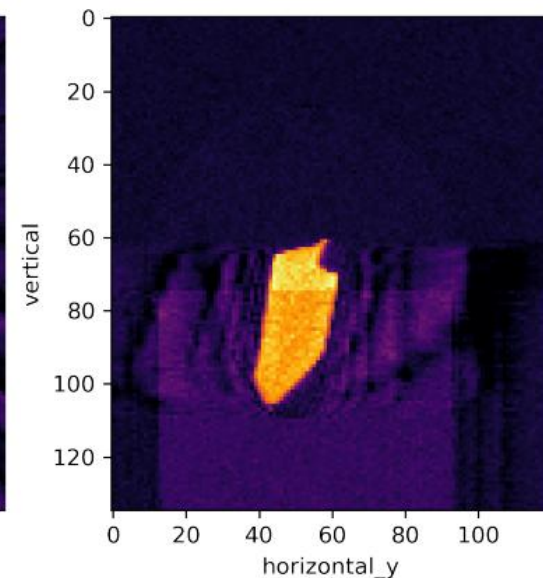
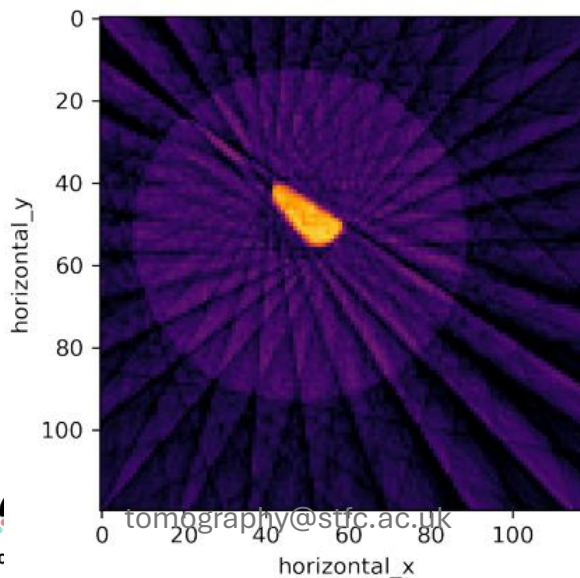


# Demonstration dataset

90  
projections



15  
projections

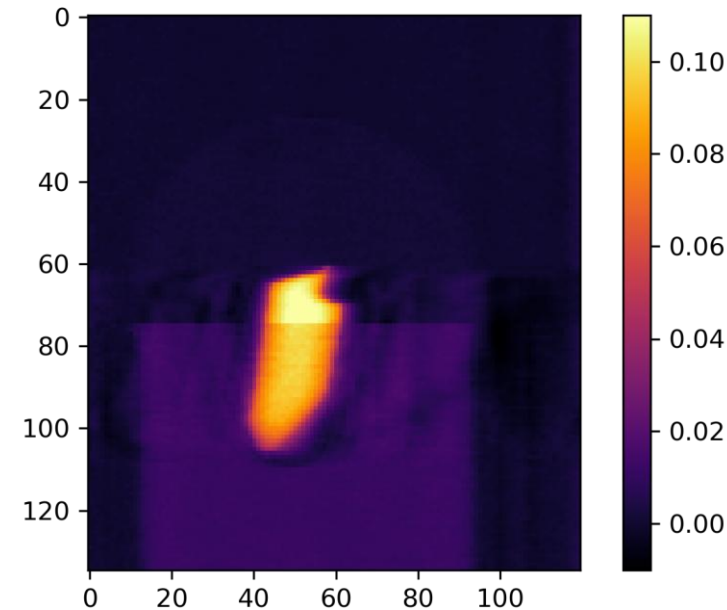
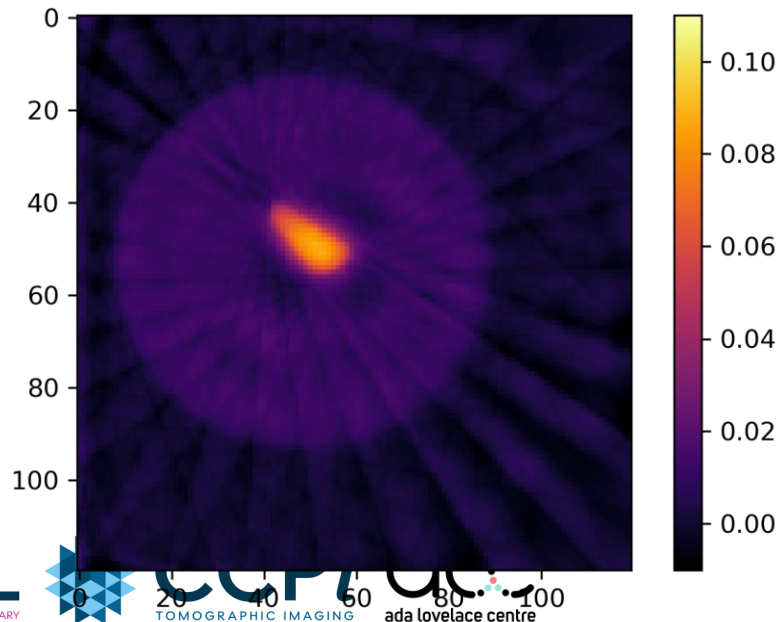


# Smooth Regularisation: Tikhonov

$$u^* = \arg \min_u \left\{ \|Au - b\|_2^2 + \alpha^2 \|Lu\|_2^2 \right\}$$

Diagram illustrating the Tikhonov regularisation equation with annotations:

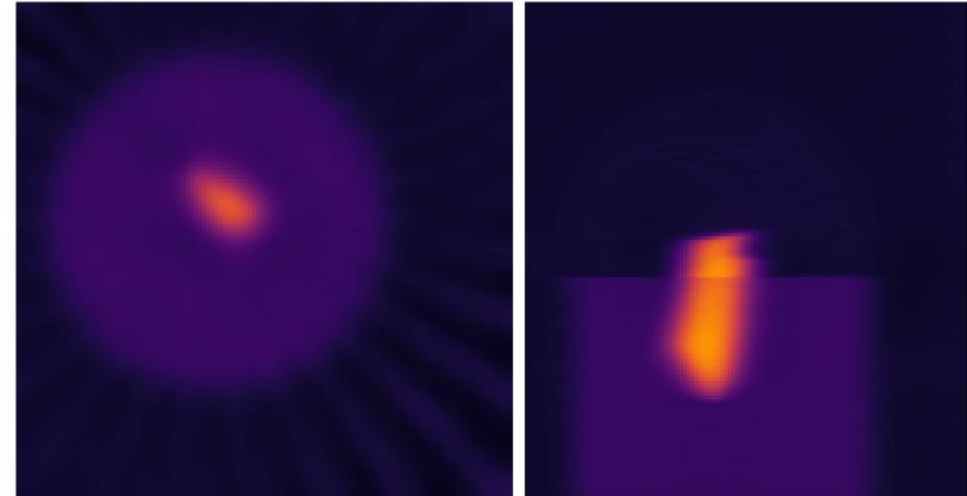
- $u^*$ : Minimiser: Solution image
- $u$ : Unknown image TBD
- $\|Au - b\|_2^2$ : Data fidelity
- $\alpha^2 \|Lu\|_2^2$ : Regulariser
- $\alpha$ : Regularisation parameter



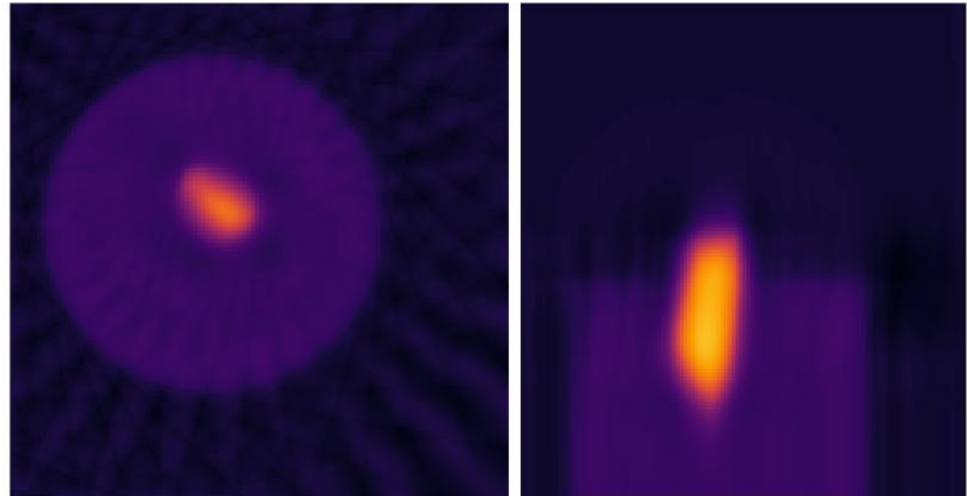
# Smooth Regularisation: Anisotropic Tikhonov

$$u^* = \arg \min_u \left\{ \|Au - b\|_2^2 + \alpha_x^2 \|L_x u\|_2^2 + \alpha_y^2 \|L_y u\|_2^2 + \alpha_z^2 \|L_z u\|_2^2 \right\}$$

Large horizontal,  
small vertical smoothing



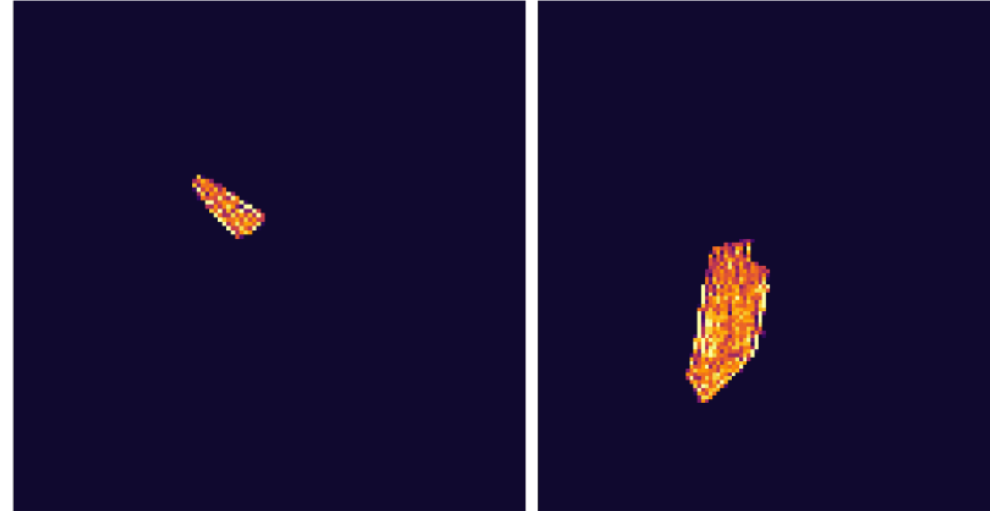
Small horizontal,  
large vertical smoothing



# Sparsity: L1 Regularization

L1-norm regularisation:

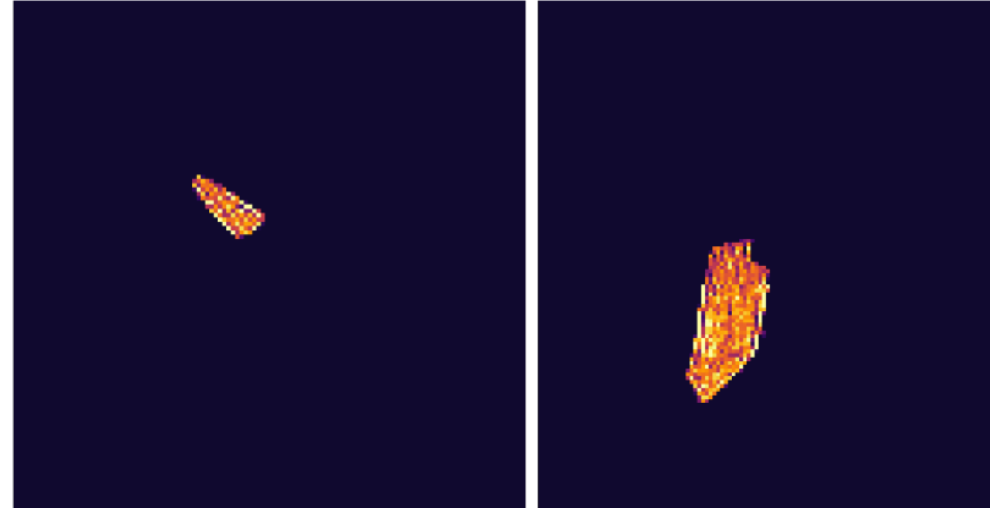
$$\|u\|_1 = \sum_j |u_j|$$



# Sparsity and Total Variation Regularization

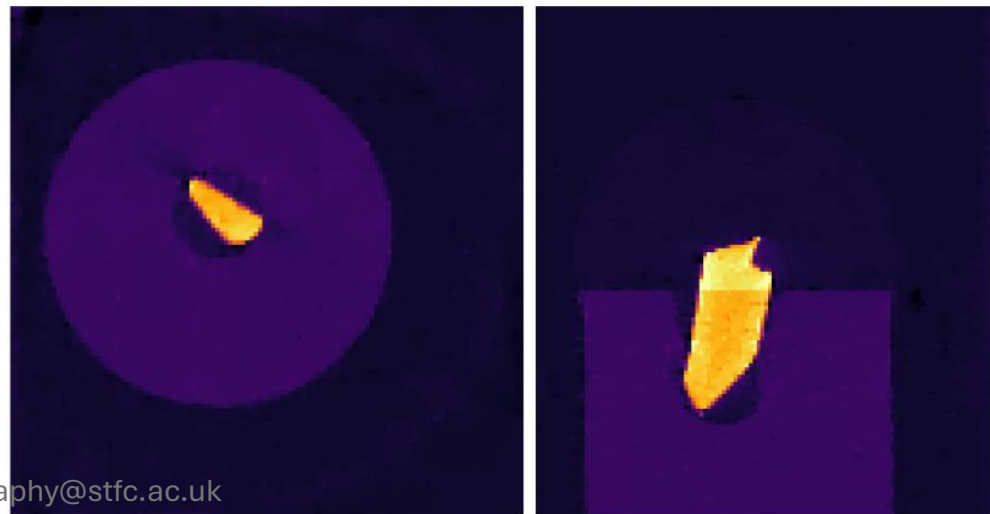
**L1-norm regularisation:**

$$\|u\|_1 = \sum_j |u_j|$$



**Total variation regularisation:**

$$\sum_j \|D_j u\|_2$$



# Regularisation demos

- Either:
  - Laminography example:

CIL-Demos/demos/2\_Iterative/05\_Laminography\_with\_TV.ipynb

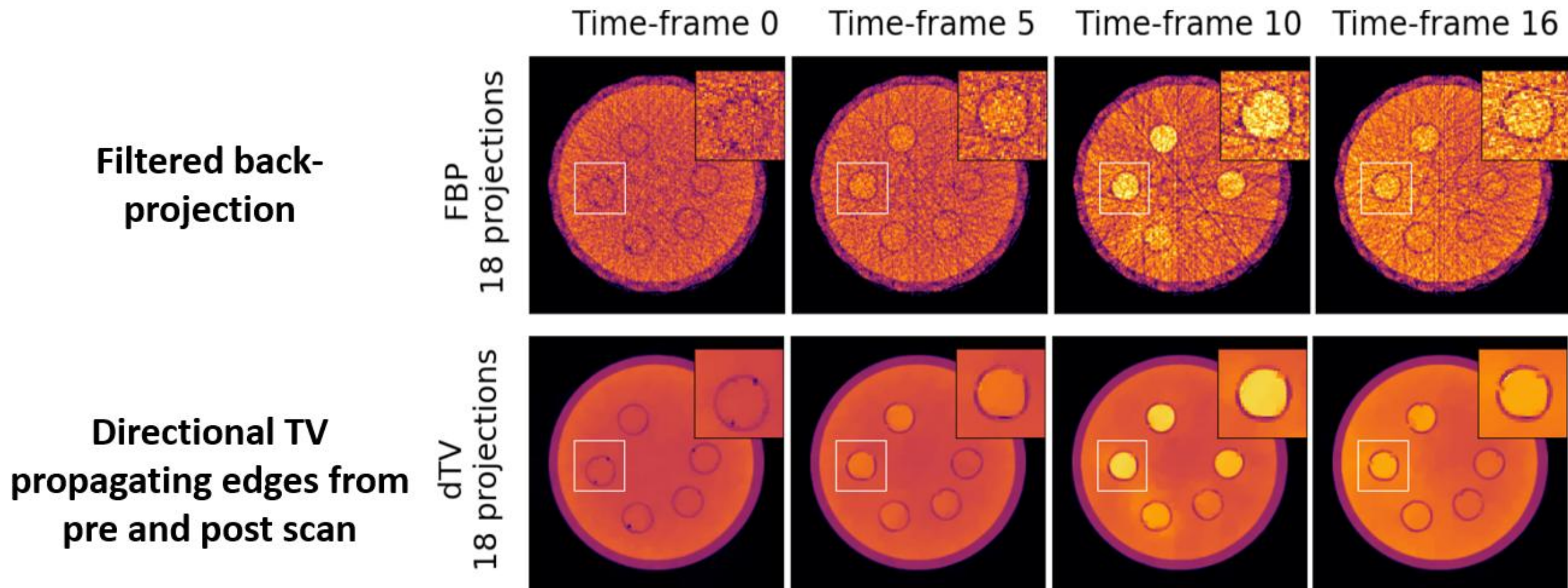
- Results of different regularisation choices

CIL-Demos/demos/2\_Iterative/01\_optimisation\_gd\_fista.ipynb

**25 minutes to run the notebook, then discussion**

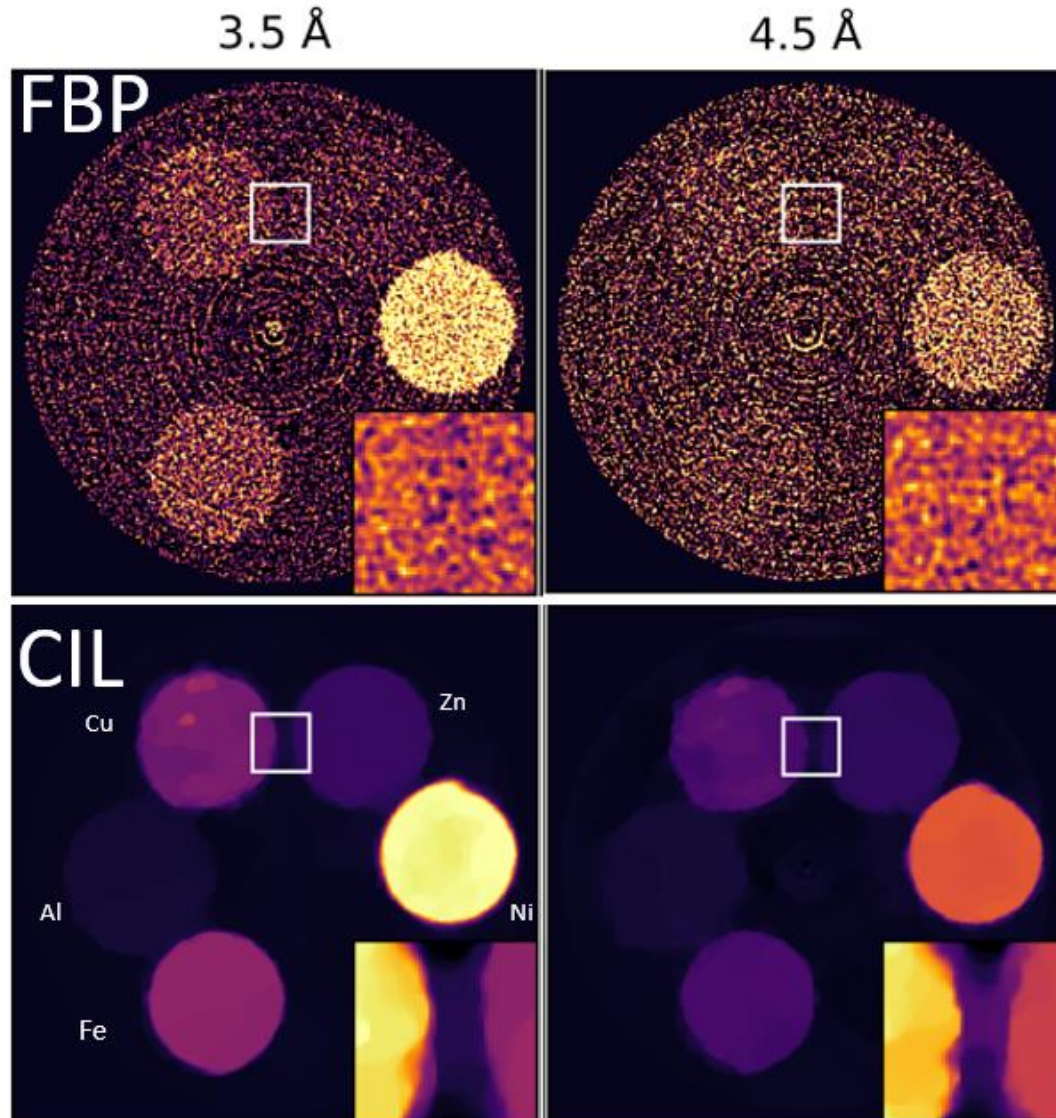


# CIL example - few-view dynamic CT

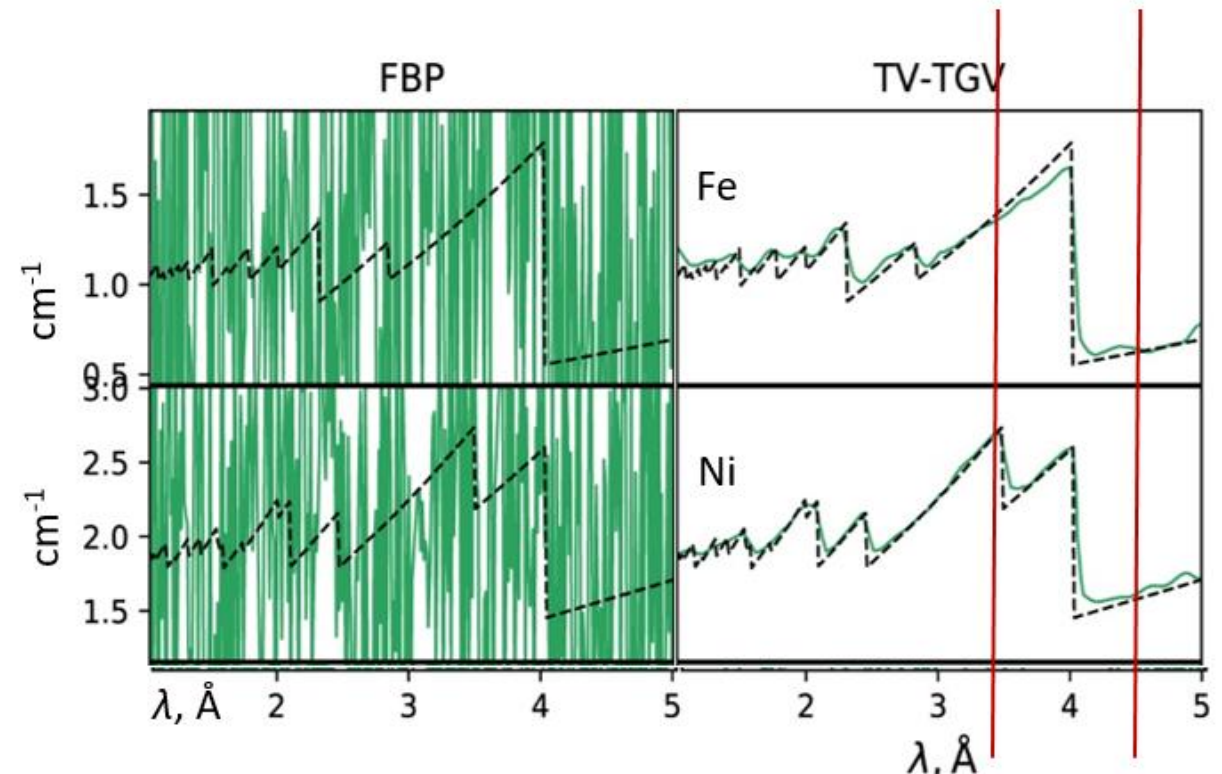


Papoutsellis et al. 2021: *Core Imaging Library - Part II: multichannel reconstruction for dynamic and spectral tomography*, Phil. Trans. R. Soc. A, **379**, 20200193: <https://doi.org/10.1098/rsta.2020.0193>

# Energy-resolved neutron CT



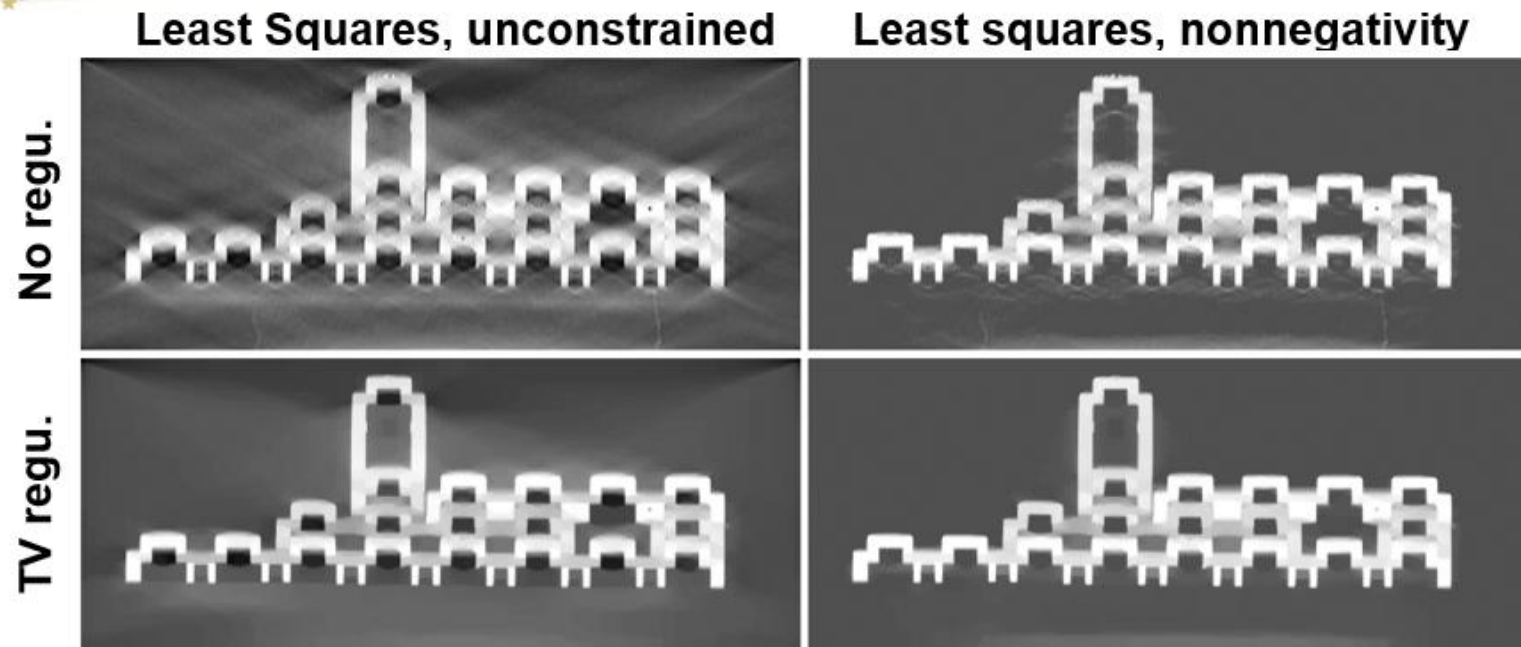
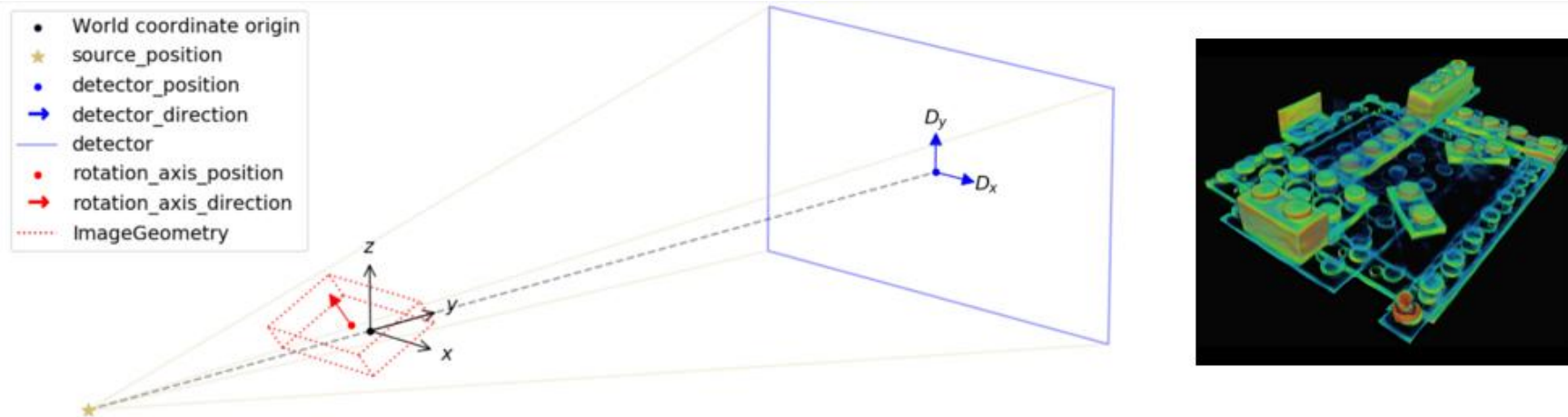
- Proposed spatio-spectral TV-TGV regularization
- Enables clear identification of Bragg edges in 3D



Ametova et al. 2021: *Crystalline phase discriminating neutron tomography using advanced reconstruction methods*, J. Physics D, <https://doi.org/10.1088/1361-6463/ac02f9>



# CIL example - non-standard scan



# CIL Optimisation module

name	description	problem type solved
CGLS	conjugate gradient least squares	least squares
SIRT	simultaneous iterative reconstruction technique	weighted least squares
GD	gradient descent	smooth
FISTA	fast iterative shrinkage-thresholding algorithm	smooth + non-smooth
LADMM	linearized alternating direction method of multipliers	non-smooth
PDHG	primal dual hybrid gradient	non-smooth
SPDHG	stochastic primal dual hybrid gradient	non-smooth

IdentityOperator		
MaskOperator	L2NormSquared	squared $L^2$ -norm: $\ x\ _2^2 = \sum_i x_i^2$
SymmetrisedGradientOperator	LeastSquares	least-squares data fidelity: $\ Ax - b\ _2^2$
ZeroOperator	MixedL21Norm	mixed $L^{2,1}$ -norm: $\ (U_1; U_2)\ _{2,1} = \ (U_1^2 + U_2^2)^{1/2}\ _1$
ProjectionOperator	SmoothMixedL21Norm	smooth $L^{2,1}$ -norm: $\ (U_1; U_2)\ _{2,1}^S = \ (U_1^2 + U_2^2 + \beta^2)^{1/2}\ _1$
ProjectionOperator	WeightedL2NormSquared	weighted squared $L^2$ -norm: $\ x\ _w^2 = \sum_i (w_i \cdot x_i^2)$

TotalVariation

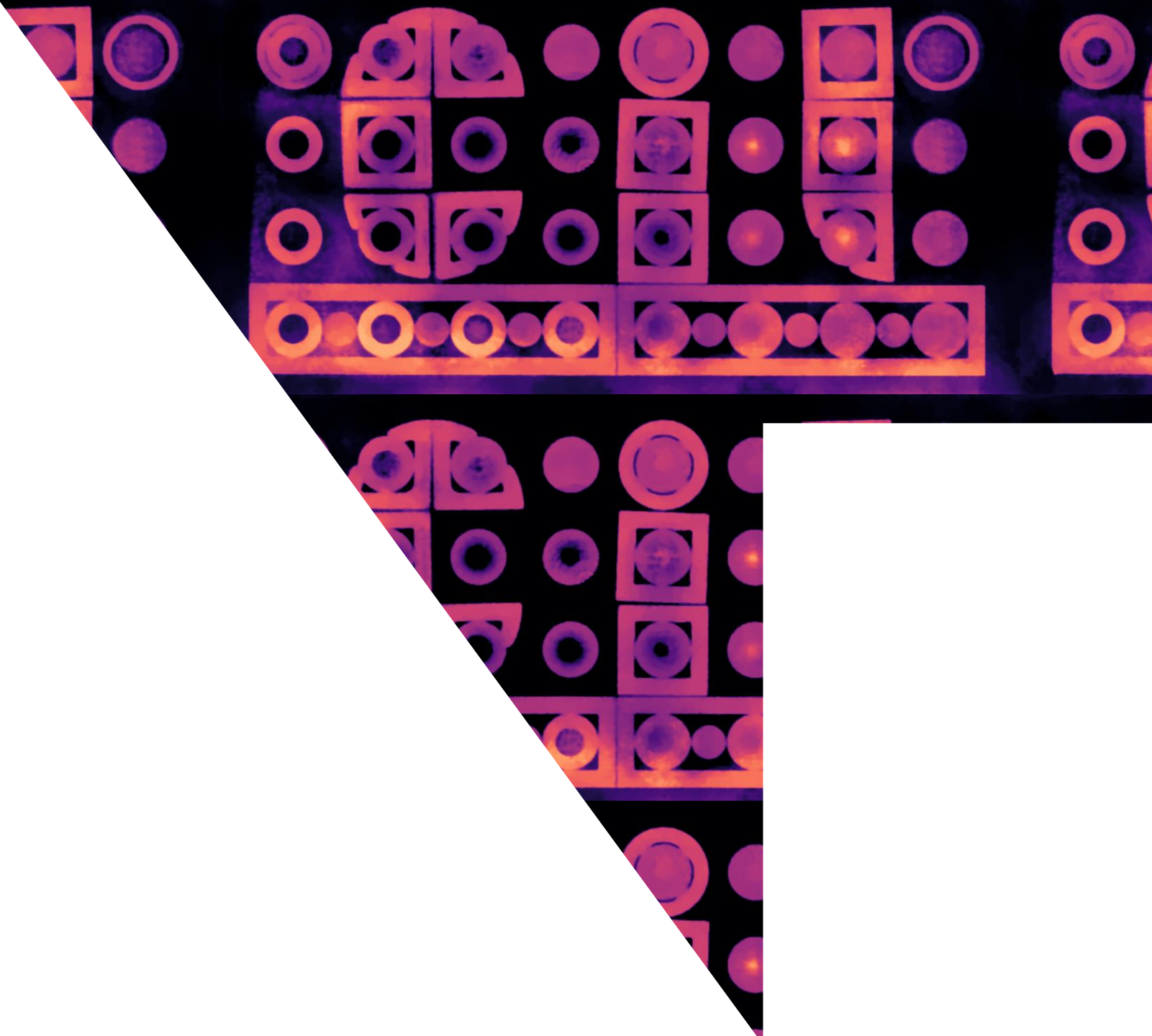
$$\text{TV}(u) = \|Du\|_{2,1} = \sum_{i,j} \left( \sqrt{(D_y u)^2 + (D_x u)^2} \right)_{i,j}$$

# Optimisation algorithms in CIL

Gradient Descent (GD)	When your objective is convex and differentiable
Conjugate Gradient Least Squares (CGLS)	For minimising a least squares problem e.g. $\min_u \ Au - b\ _2^2$
Simultaneous Iterative Reconstruction Technique (SIRT)	To solve problems of the form $Au = b$ with optional constraints
Iterative Shrinkage-Thresholding Algorithm (ISTA)	To solve problems of the form $\min_u f(u) + g(u)$ where $f$ is convex and differentiable and $g$ is convex with a simple proximal operator
Fast Iterative Shrinkage-Thresholding Algorithm (FISTA)	Like ISTA but accelerated
Primal Dual Hybrid Gradient (PDHG)	To solve problems of the form $\min_u f(Au) + g(u)$ where $f$ is convex and has a “simple” proximal method of its conjugate and $g$ is convex with a “simple” proximal.
Stochastic Primal Dual Hybrid Gradient (SPDHG)	Similar to PDHG but where $f$ can be written as a separable sum
Linearized Alternating Direction Method of Multipliers (LADMM)	To solve problems of the form $\min_u f(u) + g(v)$ subject to $Au + Bv = b$ where both $f$ and $g$ are convex
Stochastic algorithms...	Training coming soon...

$$\text{prox}_{\tau g}(u) = \arg \min_v \left\{ \tau g(v) + \frac{1}{2} \|v - u\|_2^2 \right\}$$

**Find out more**



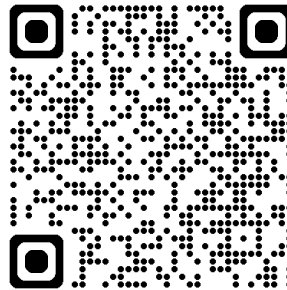
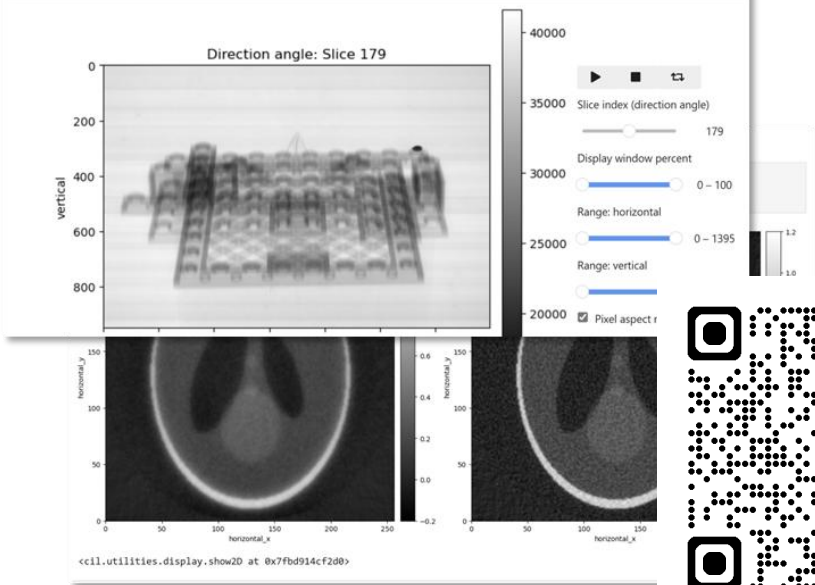
# Demos and examples

21+ demo notebooks

 [TomographicImaging/CIL-Demos](https://github.com/TomographicImaging/CIL-Demos)

As we have already defined our acquisition geometry we can use the function `read_as_AcquisitionData()` to pass this to the reader. The reader will use this to configure and return an `AcquisitionData` object containing the data and the geometry describing it.

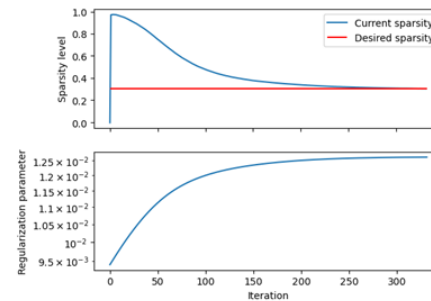
```
path = 'Lego_Lamino30deg_XTH/'  
  
reader = TIFFStackReader(file_name=os.path.join(path_common, path), roi=roi, mode='slice')  
acq_data_raw = reader.read_as_AcquisitionData(ag)  
  
islicer(acq_data_raw, direction='angle', origin='upper-left')
```



14 contributed notebooks showcasing interesting uses of CIL

 [TomographicImaging/CIL-User-Showcase](https://github.com/TomographicImaging/CIL-User-Showcase)

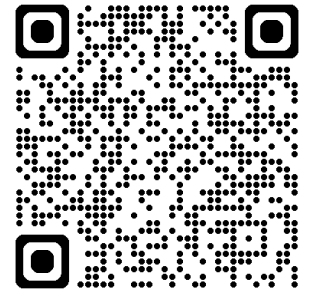
```
In [28]: import matplotlib.pyplot as plt  
  
N = fista_controlled.iterations[-1]  
fig, ax = plt.subplots(2)  
ax[0].plot(sparsity_ctrl.sparsity_vals, label='Current sparsity')  
ax[0].hlines(sparsity_ctrl.desired_sparsity, xmin=0, xmax=N, color='r', label='Desired sparsity')  
ax[0].set_ylabel('Sparsity level')  
ax[0].legend()  
  
ax[1].semilogy(sparsity_ctrl.reg_param_vals)  
ax[1].set_ylabel('Regularization parameter')  
ax[1].set_xlabel('Iteration')  
for a in ax.flat:  
    a.label_outer()
```



Here we can see how increasing the regularization parameter (lower plot) drives the sparsity level down towards the desired level (top plot). The iteration can be stopped once the suitable level has been reached (and the relative change is small enough).

Notice also how the tuning is adaptive, in the sense that as we get closer to the desired sparsity level, the incremental changes in the regularization parameter get finer.

Adjusting the contrast we can see there still remains a bright circular artifact in the horizontal slice and a stripe p.





# CIL Publications

Jørgensen et al.: *Core Imaging Library - Part I: a versatile Python framework for tomographic imaging* Phil. Trans. R. Soc. A. **379** 20200192 (2021) DOI: [10.1098/rsta.2020.0192](https://doi.org/10.1098/rsta.2020.0192)

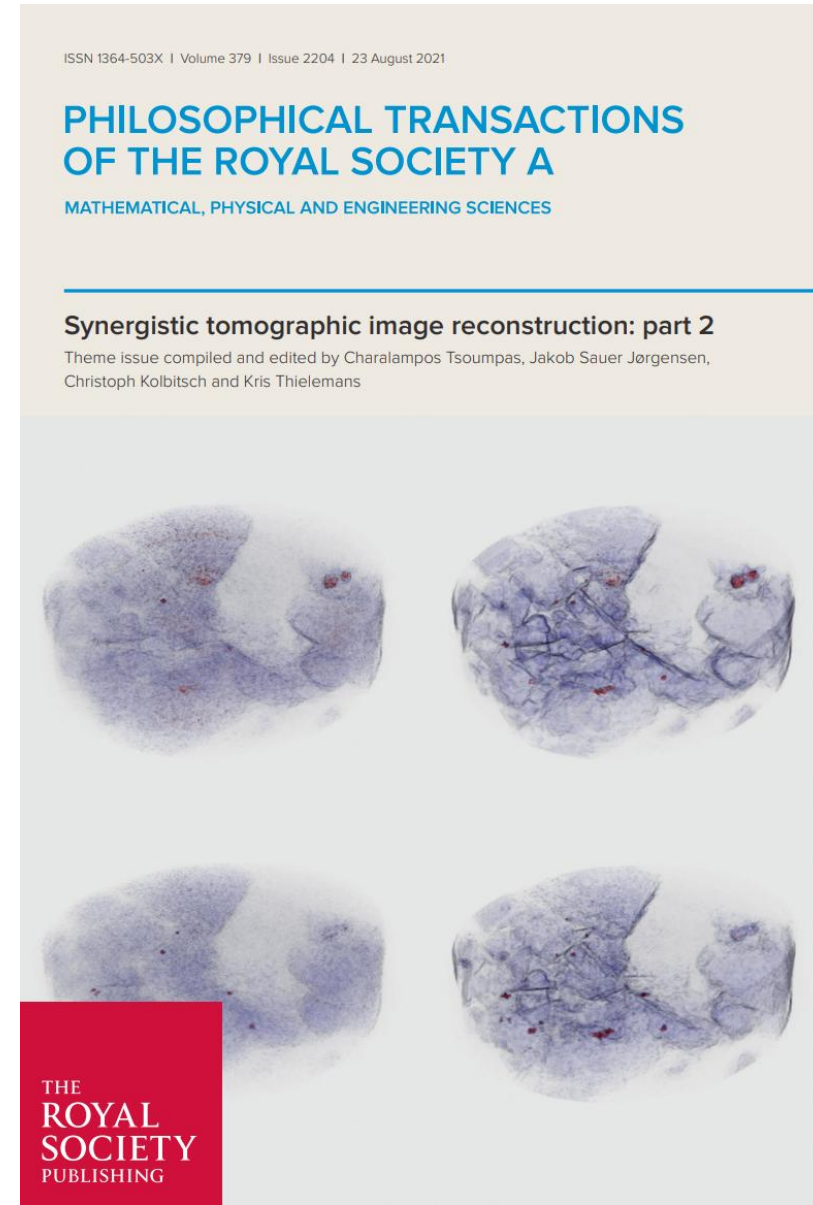
Papoutsellis et al.: *Core Imaging Library - Part II: multichannel reconstruction for dynamic and spectral tomography* Phil. Trans. R. Soc. A. **379** 20200193 (2021) DOI: [10.1098/rsta.2020.0193](https://doi.org/10.1098/rsta.2020.0193)

Jørgensen et al.: *A directional regularization method for the limited-angle Helsinki Tomography Challenge using the Core Imaging Library (CIL)*, Applied Mathematics for Modern Challenges, Volume **1**, Issue 2: 143-169. (2023) [10.3934/ammc.2023011](https://doi.org/10.3934/ammc.2023011)

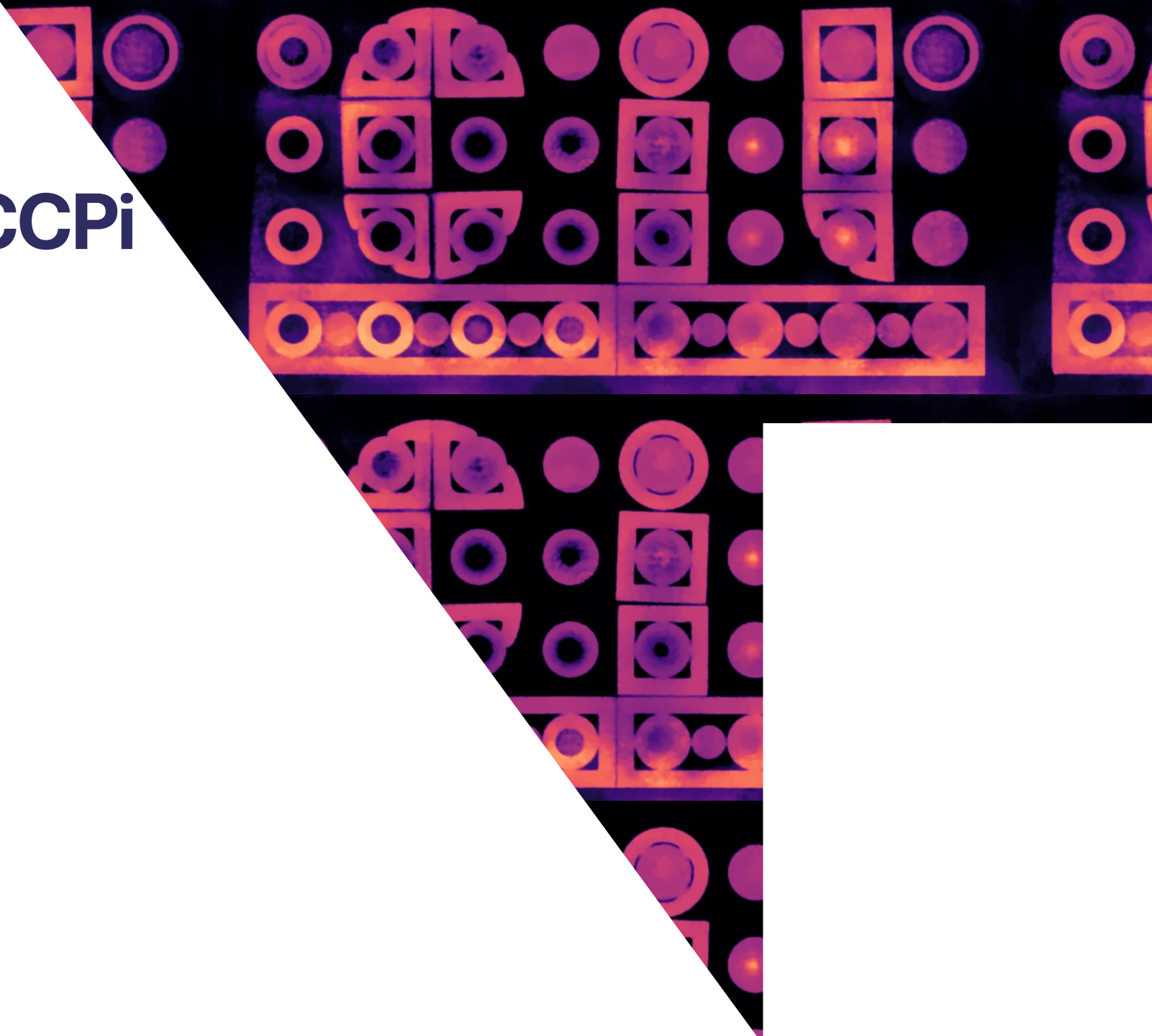
Ametova et al.: *Crystalline phase discriminating neutron tomography using advanced reconstruction methods*, J. Phys. D: Appl. Phys. **54** 325502 (2021) DOI [10.1088/1361-6463/ac02f9](https://doi.org/10.1088/1361-6463/ac02f9)

Warr R. et al.: *Enhanced hyperspectral tomography for bioimaging by spatio-spectral reconstruction* Sci Rep **11**, 20818 (2021) DOI: [10.1038/s41598-021-00146-4](https://doi.org/10.1038/s41598-021-00146-4)

Brown R. et al.: *Motion estimation and correction for simultaneous PET/MR using SIRT and CIL* Phil. Trans. R. Soc. A. **379** 20200208 (2021) DOI: [10.1098/rsta.2020.0208](https://doi.org/10.1098/rsta.2020.0208)



# The CIL and Wider CCPi Community



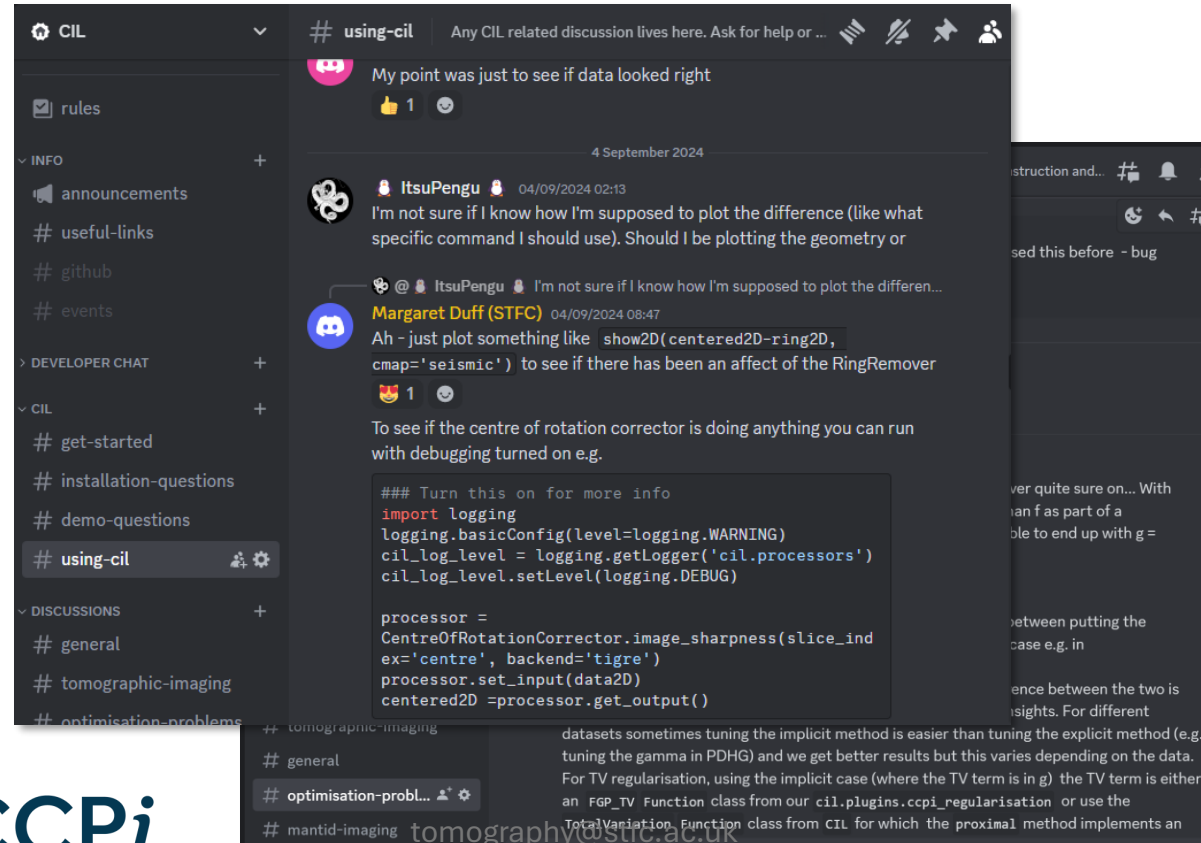
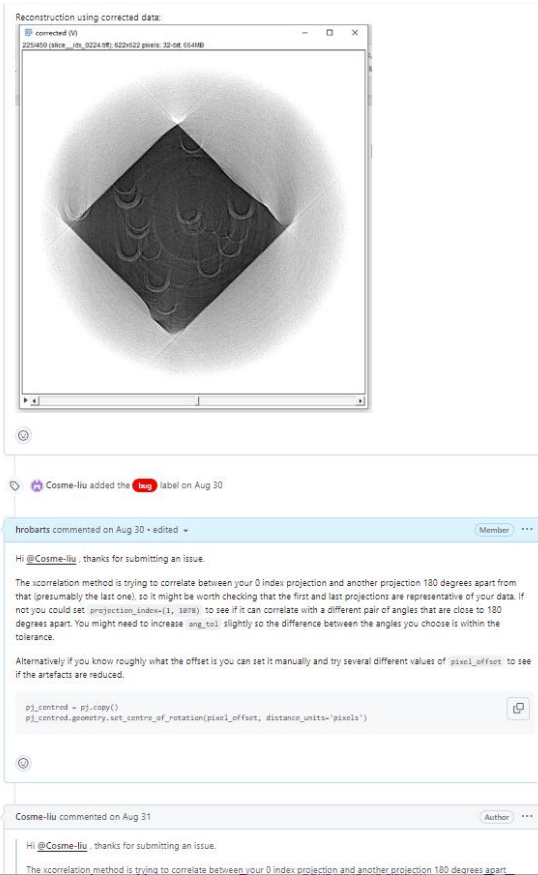
# CIL User Community

User support  
through GitHub

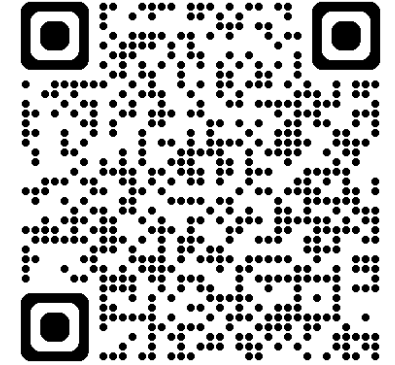
And CIL discord  
with 200+ users

General and  
tailored support

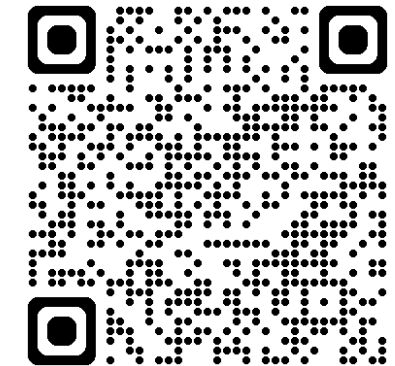
Fortnightly User  
Drop-ins



 CIL Discord



 CIL GitHub





# CIL User Community Events

Travel grants

In person and  
online training

Annual User Meeting

Data and software  
hackathons

Fortnightly Show  
and Tell Meetings

For more details...  
<https://ccpi.ac.uk>





# CIL User meeting

- 27th-30th January
- Rutherford Appleton Laboratory, Oxfordshire, UK.

An opportunity to meet others using CIL, share your work, and help shape its future

- Talks from the CIL community
- Training and Hackathon
- Poster session and networking opportunities

For more information:

<https://ccpi.ac.uk/events/cil-um-26/>





# CIL Data Hackathon

- 18th-20th November
- Rutherford Appleton Laboratory, Oxfordshire, UK
- Daresbury Laboratory, Cheshire, UK

Help us create a library of example readers showcasing the use of each reader on open-access CT datasets.

For more information:

<https://ccpi.ac.uk/events/cil-ct-data-reader-hackathon/>

# Wrap up

- CIL is an Open Source mostly Python library for all your tomographic needs:
  - I/O
  - pre-processing
  - Reconstruction
  - Visualisation
- Developer Support, user driven, long term funding
- Join the community Discord
- <https://www.ccpi.ac.uk/CIL>