Optimisation-based reconstruction and Core Imaging Library examples

Jakob Sauer Jørgensen

Senior Researcher, DTU Compute

DTU course 02946 – week 2, day 4
12 January 2023

Main take-away messages

Filtered back-projection is pretty darn good!
If data is good, look no further.

If data is bad, iterative reconstruction methods *may* help...
Different kinds of data need different methods.

Core Imaging Library (CIL) is an open-source Python package providing a wide range of iterative tomographic reconstruction methods.

Iterative reconstruction methods (algebraic methods)

The SIRT algorithm

Simultaneous Iterative Reconstruction Technique (SIRT):

$$u^{(k+1)} = u^{(k)} + t_k C A^T R \left(b - A u^{(k)} \right), \qquad k = 0, 1, 2, \ldots$$

where for matrix $A \in \mathbb{R}^{m \times n}$:

$$R \in \mathbb{R}^{m \times m} \quad \text{with} \quad r_{ii} = 1/\sum_j a_{ij} \quad \text{(inverse row sums)}$$

$$C \in \mathbb{R}^{n \times n} \quad \text{with} \quad c_{jj} = 1/\sum_i a_{ij} \quad \text{(inverse column sums)}$$

Properties:

- Faster than Landweber.
- Step size selection easier: Converges for $0 < t_k < 2$.
- Balances short/long rays by weighted least-squares:

$$\min_u \|b - Au\|_R \quad \text{where} \|u\|_R = u^T R u$$

- Can incorporate constraints, such as nonnegativity $u \geq 0$.

The CGLS algorithm

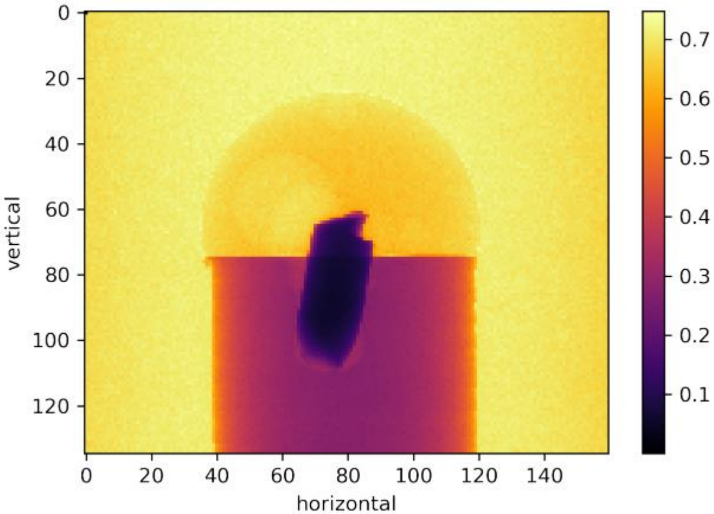
Conjugate Gradient Least Squares (CGLS):

- Krylov subspace method for least squares $\min_u \|Au - b\|_2$.
- Unlike SIRT which only uses current image, CGLS uses images from all previous iterations to determine next step.
- Typically much faster than SIRT (still semi-convergence).
- Constraints not possible.
- Pseudo code:

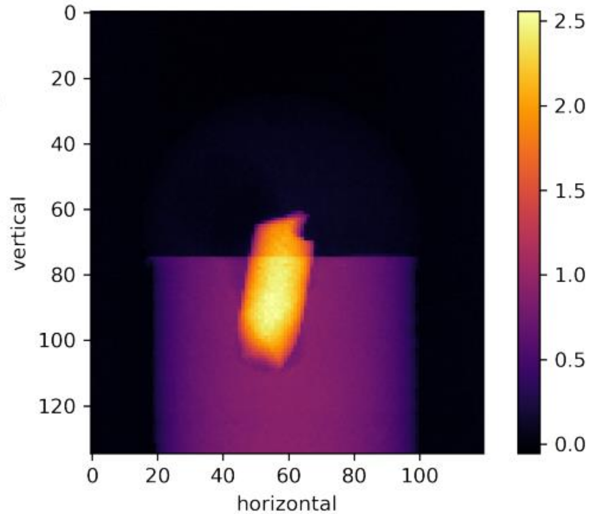
$$u^0 = \text{initial vector}$$
$$r^0 = b - Au^0$$
$$d^0 = A^T r^0$$
for $k = 1, 2, \ldots, \text{maxiter}$
$$\bar{\alpha}_k = \|A^T r^{k-1}\|_2^2 / \|Ad^{k-1}\|_2^2$$
$$u^k = u^{k-1} + \bar{\alpha}_k d^{k-1}$$
$$r^k = r^{k-1} - \bar{\alpha}_k A d^{k-1}$$
$$\bar{\beta}_k = \|A^T r^k\|_2^2 / \|A^T r^{k-1}\|_2^2$$
$$d^k = A^T r^k + \bar{\beta}_k d^{k-1}$$
end

Demonstration data set

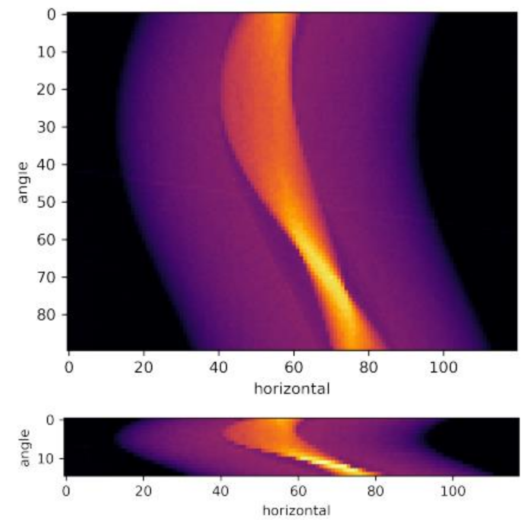
Raw projection



Negative log, cropped, centered



Sinogram



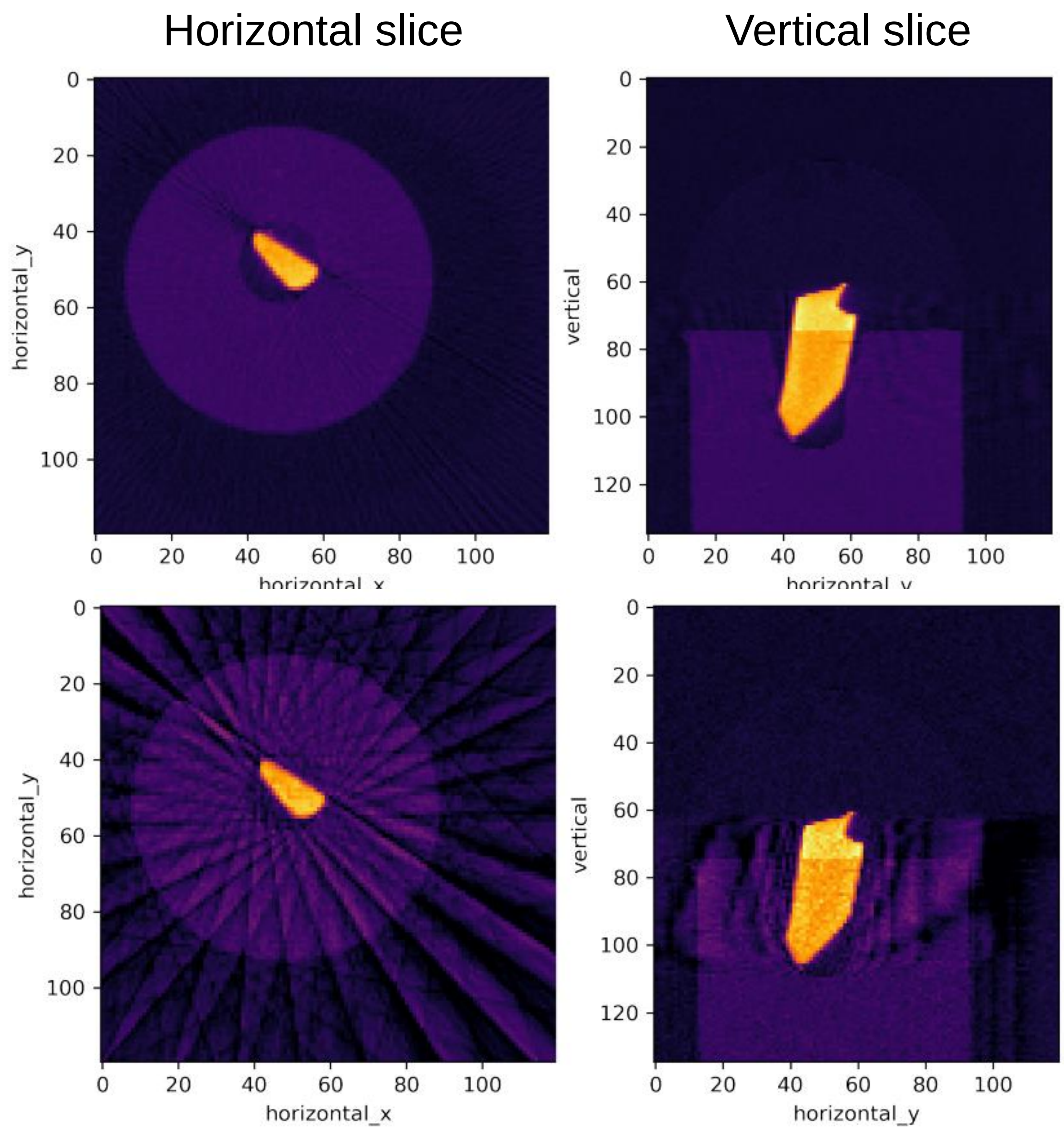
- 3D parallel-beam data set from Diamond Light Source, UK
- 0.5mm aluminium cylinder with piece of steel wire
- Droplet salt water causing corrosion + hydrogen bubbles
- Part of a fast time-lapse experiment
- 90 projections over 180 degrees, and **15 projections**
- Downsampled to 160-by-135 pixels for quick demonstration

J. et al. 2021: *Core Imaging Library - Part I: a versatile Python framework for tomographic imaging,* Phil Trans A. https://doi.org/10.1098/rsta.2020.0192

Filtered backprojection

90 projections

15 projections



Algebraic iterative methods (regularizing by number of iterations)

CGLS

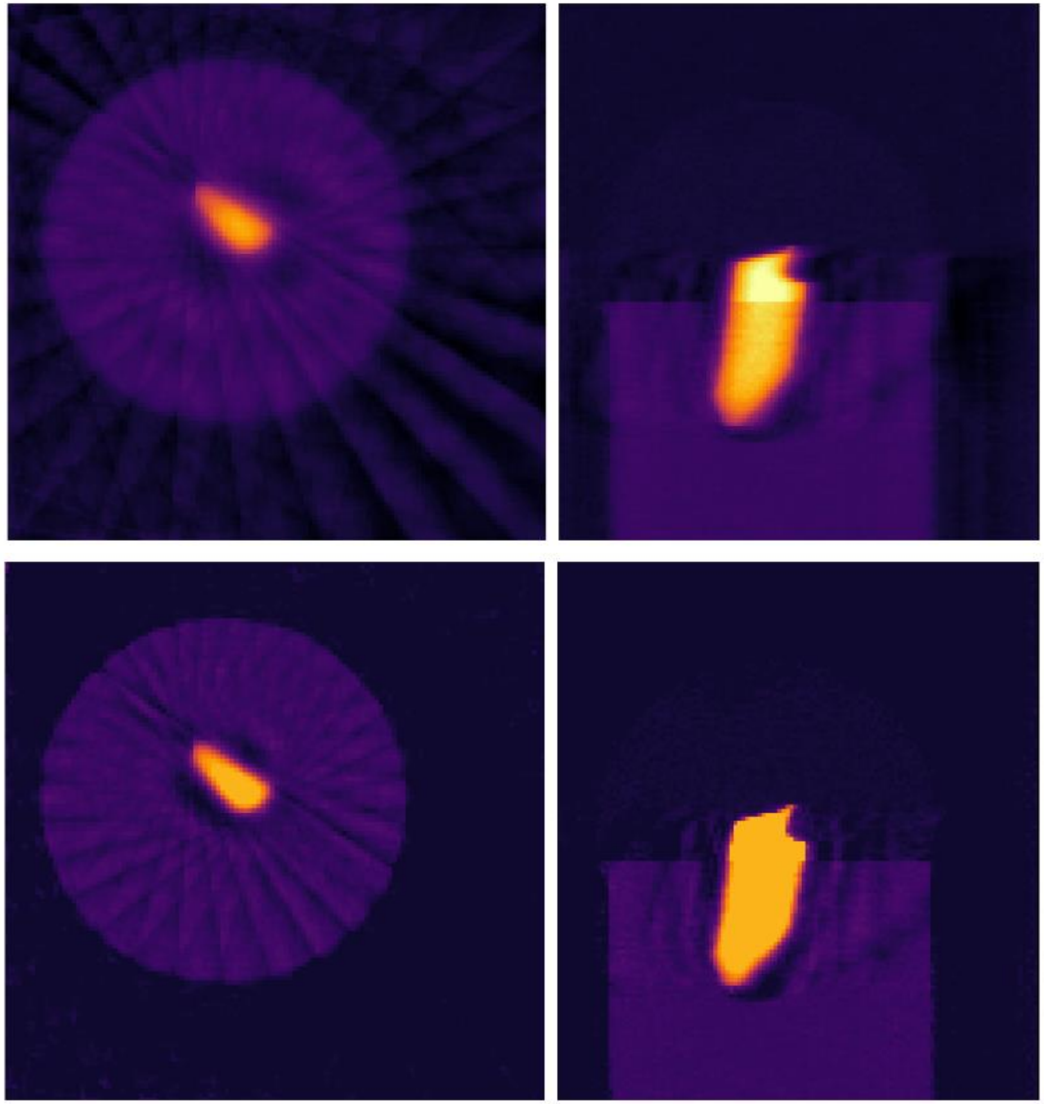
$$u^\star = \arg\min_u ||Au - b||_2^2$$

Typically 10s of iterations

SIRT

As above and allowing lower and upper bounds on pixel values, here Non-negative and <= 0.9

Typically 100s of iterations



Algebraic methods – pros and cons

The good:

- Flexible geometry: No specific inversion formulas required.
- Can incorporate simple constraints, e.g., nonnegativity.
- Sometimes give better reconstructions than FBP.

The bad:

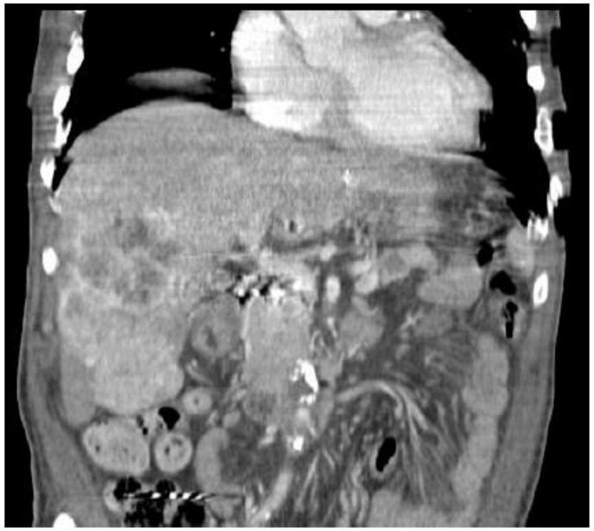
- Much more expensive than FBP:
  - 100s of iterations each of 1 forward and 1 back-projection
  - FBP: Single back-projection
- Problem of choosing when to stop.

Optimization-based iterative reconstruction methods

Iterative reconstruction is based on optimization problems and algorithms

Discrete imaging model:

$$Ax = b$$



Typical CT images:

- Regions of homogeneous tissue.
- Separated by sharp boundaries.

Reconstruction by regularization:

$$x^\star = \underset{x}{\operatorname{argmin}} \; \underbrace{\mathcal{D}(Ax, b)}_{\text{data fidelity}} + \lambda \cdot \underbrace{\mathcal{R}(x)}_{\text{regularizer}}$$

Most basic optimization problem (no regularization):

$$u^\star = \arg\min_u \|Au - b\|_2^2 = \sum_i ((Au)_i - b_i)^2$$

Regularised reconstruction – example Tikhonov

Data fidelity

Regulariser

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

Minimiser:
Solution image

Unknown
image TBD

Regularisation
parameter

- Balance between fitting data and penalizing ”large values of *Lu*”
- Different choices of *L*:
  - Identity operator – make pixel values small
  - Finite difference gradient operator – make neighbor pixels similar
- No one best regulariser - different image types need different regularisers!

Regularised reconstruction – example Tikhonov

Data fidelity

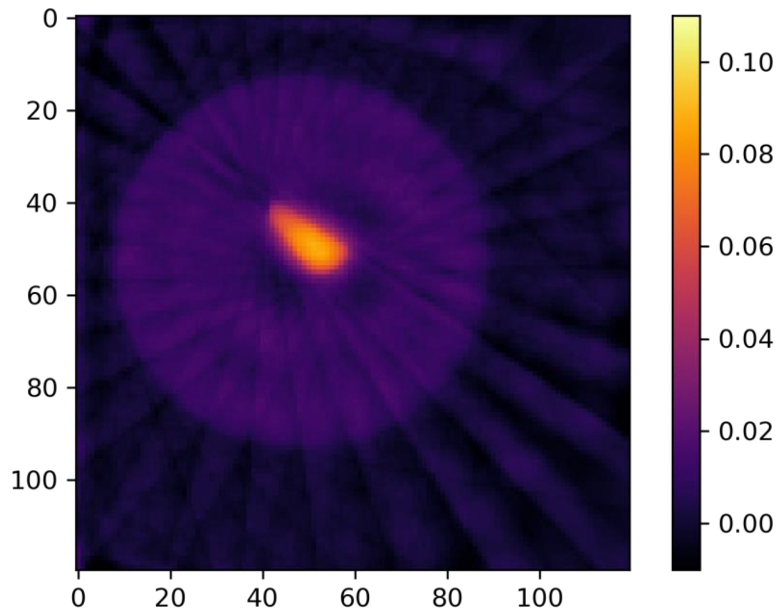
Regulariser

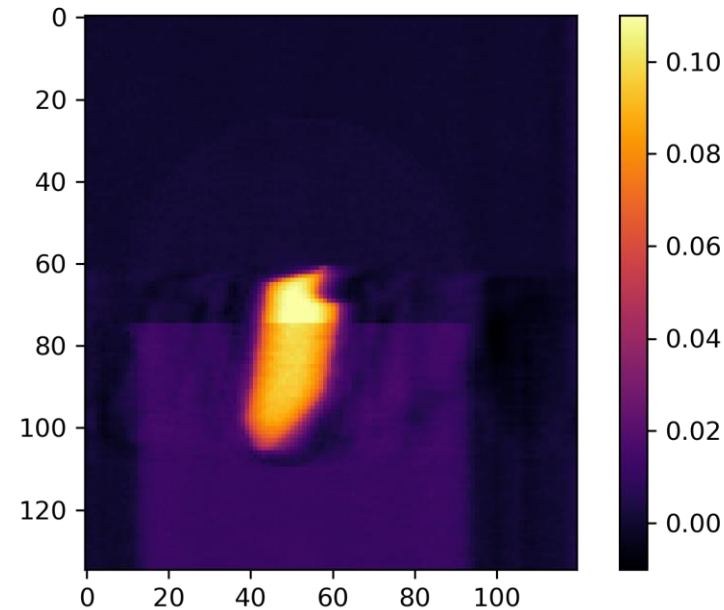
$$u^\star = \arg\min_u \left\{ \|Au - b\|_2^2 + \alpha^2 \|Lu\|_2^2 \right\}$$

Minimiser:
Solution image

Unknown
image TBD

Regularisation
parameter



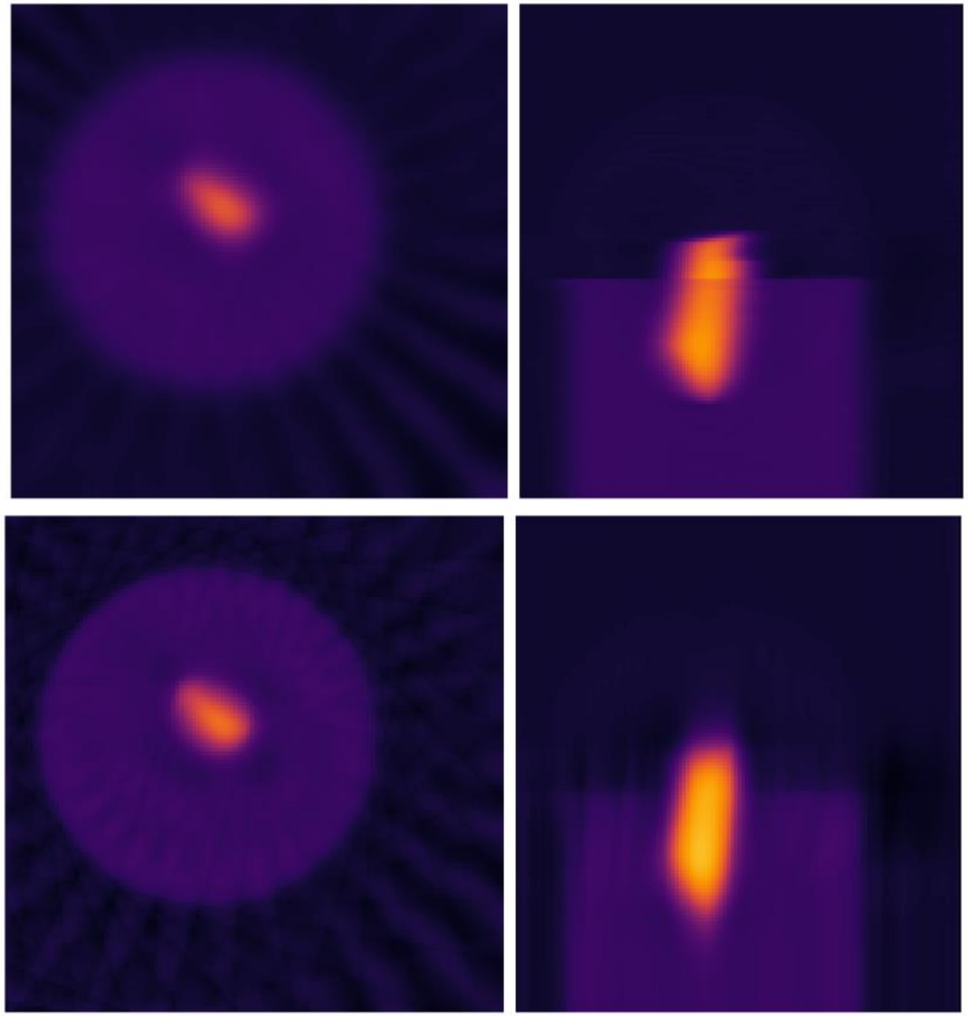


Anisotropic Tikhonov regularization

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

**Large horizontal,
small vertical smoothing**

**Small horizontal,
large vertical smoothing**



How to solve Tikhonov problem in CIL

$$\min_u \|Au - b\|^2 + \alpha^2 \|Lu\|^2$$

$$\min_u \left\| \begin{pmatrix} A \\ \alpha L \end{pmatrix} u - \begin{pmatrix} b \\ 0 \end{pmatrix} \right\|^2$$

$$\min_u \|\tilde{A}u - \tilde{b}\|^2$$

with

$$\tilde{A} = \begin{pmatrix} A \\ \alpha L \end{pmatrix}$$

$$\tilde{b} = \begin{pmatrix} b \\ 0 \end{pmatrix}$$

CGLS with $\tilde{A}$ and $\tilde{b}$

See CIL notebook:
2_Iterative/02_tikhonov_block_framework.ipynb

Gradient descent algorithm (when differentiable)

Gradient (*f* must be differentiable):

$$\nabla f(u) = \begin{pmatrix} \frac{\partial f(u)}{\partial u_1} \\ \frac{\partial f(u)}{\partial u_2} \\ \vdots \\ \frac{\partial f(u)}{\partial u_n} \end{pmatrix}$$

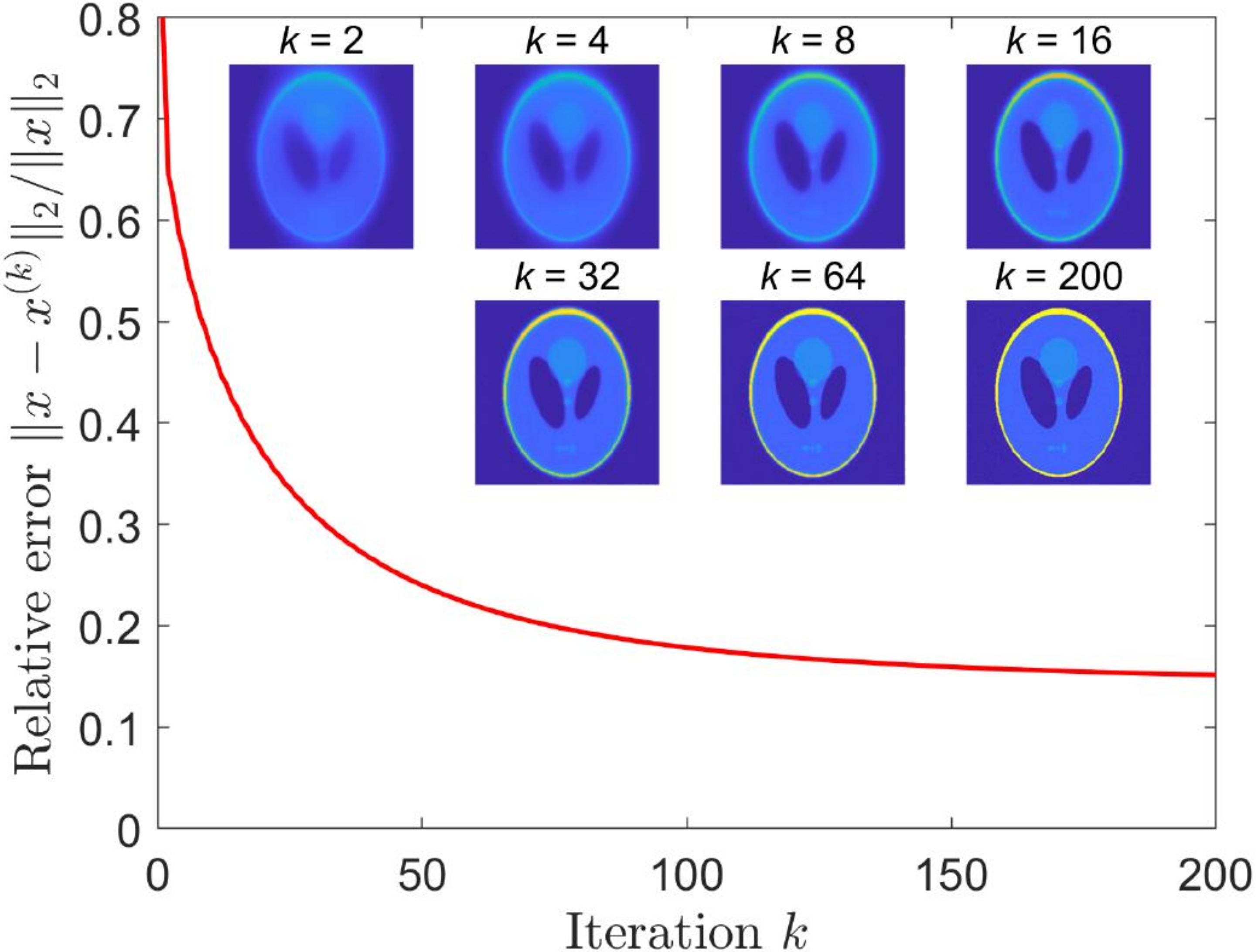
More general optimization problem

$$\arg\min_u f(u)$$

Gradient descent algorithm:

$$u^{(k+1)} = u^{(k)} - t_k \nabla f(u^{(k)}), \quad k = 0, 1, 2, \ldots$$

Gradient descent algorithm example: least squares minimization (Shepp-Logan)



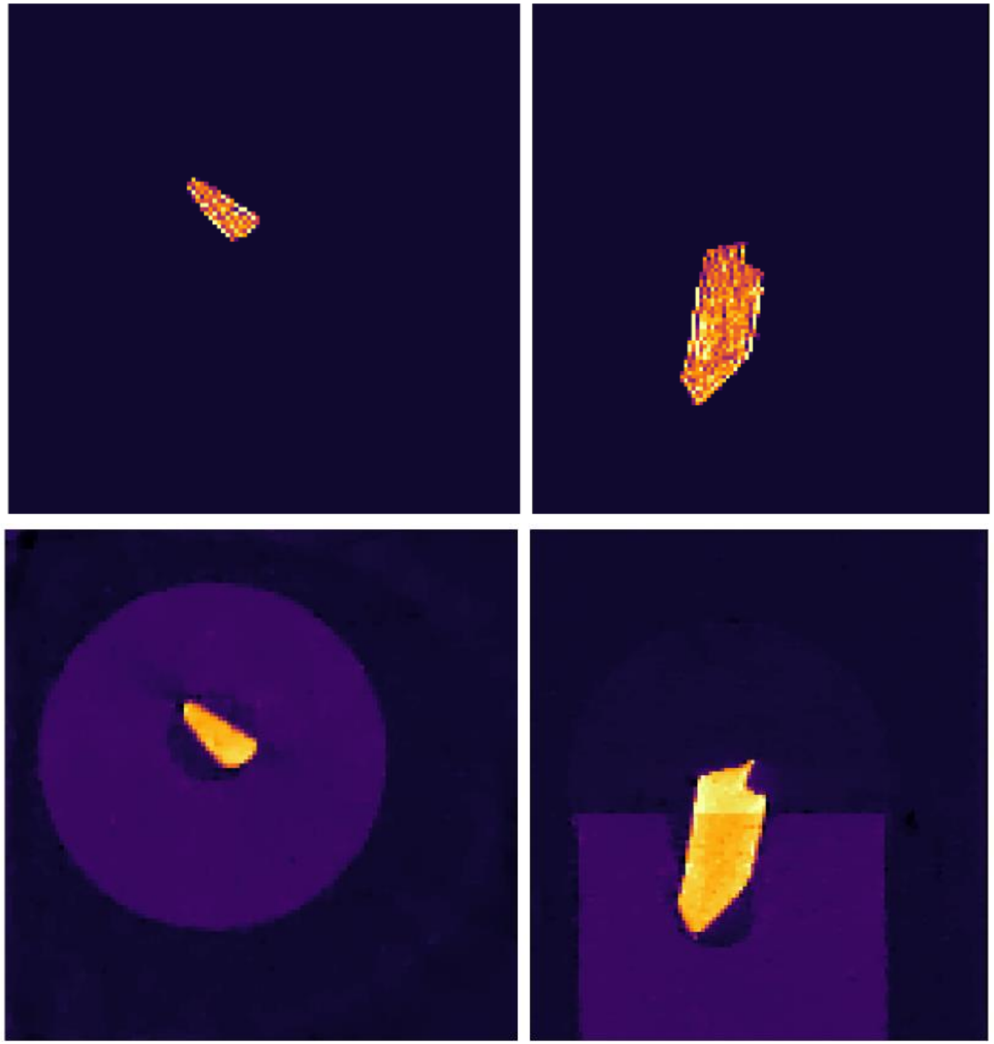
Sparsity and total variation regularization

L1-norm regularization:

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

Total variation regularization:

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



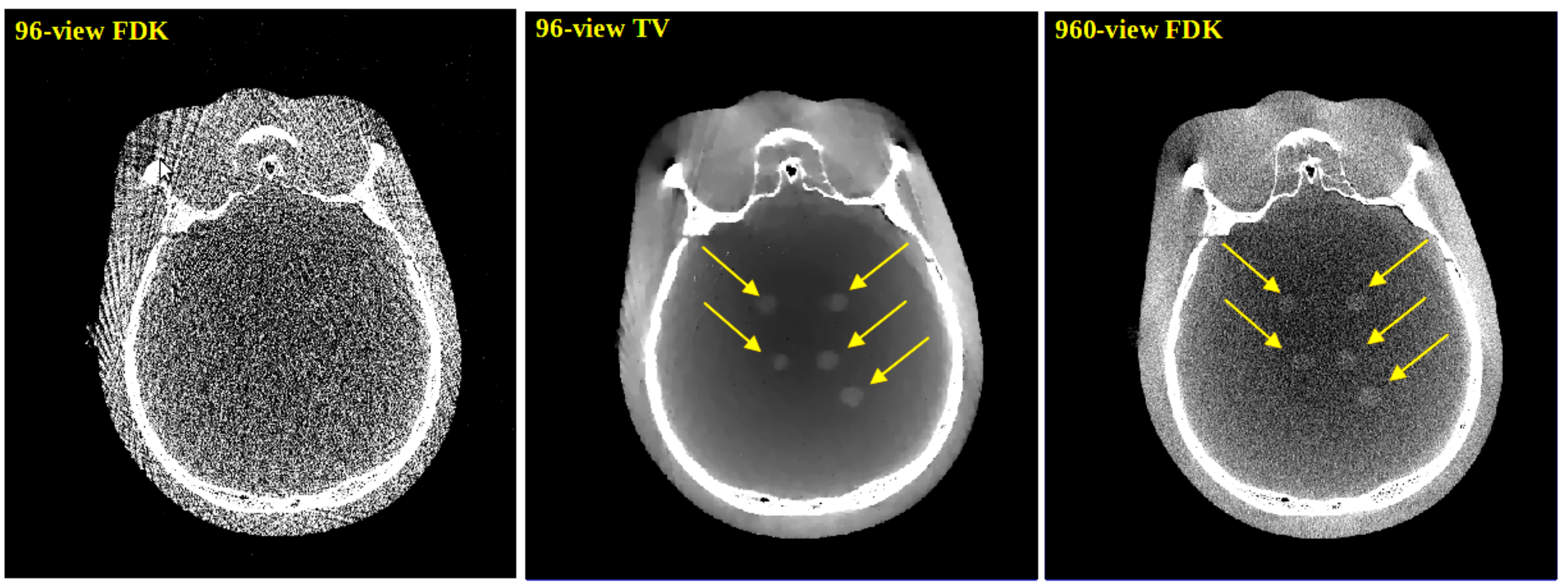
TV reconstruction example, physical head phantom, cone-beam X-ray CT

Total variation: Homogeneous regions with sharp boundaries.

$$x^\star = \underset{x}{\operatorname{argmin}} \left\{ \|Ax - b\|_2^2 + \alpha \|x\|_{\mathsf{TV}} \right\}$$

$$\|x\|_{\mathsf{TV}} = \sum_j \|D_j x\|_2, \quad D_j \text{ finite diff. gradient at voxel } j.$$

TV is an example of **sparsity-regularized reconstruction.**

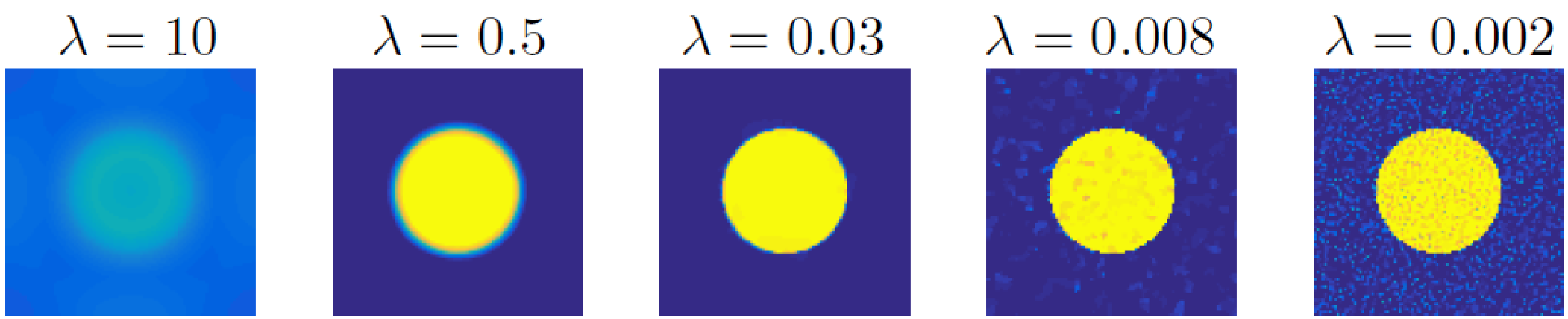


[Bian et al. 2010, Phys. Med. Biol. **55**, 6575–6599]. Courtesy: X. Pan, U. Chicago.

Effect of regularisation parameter

Total variation regularization:

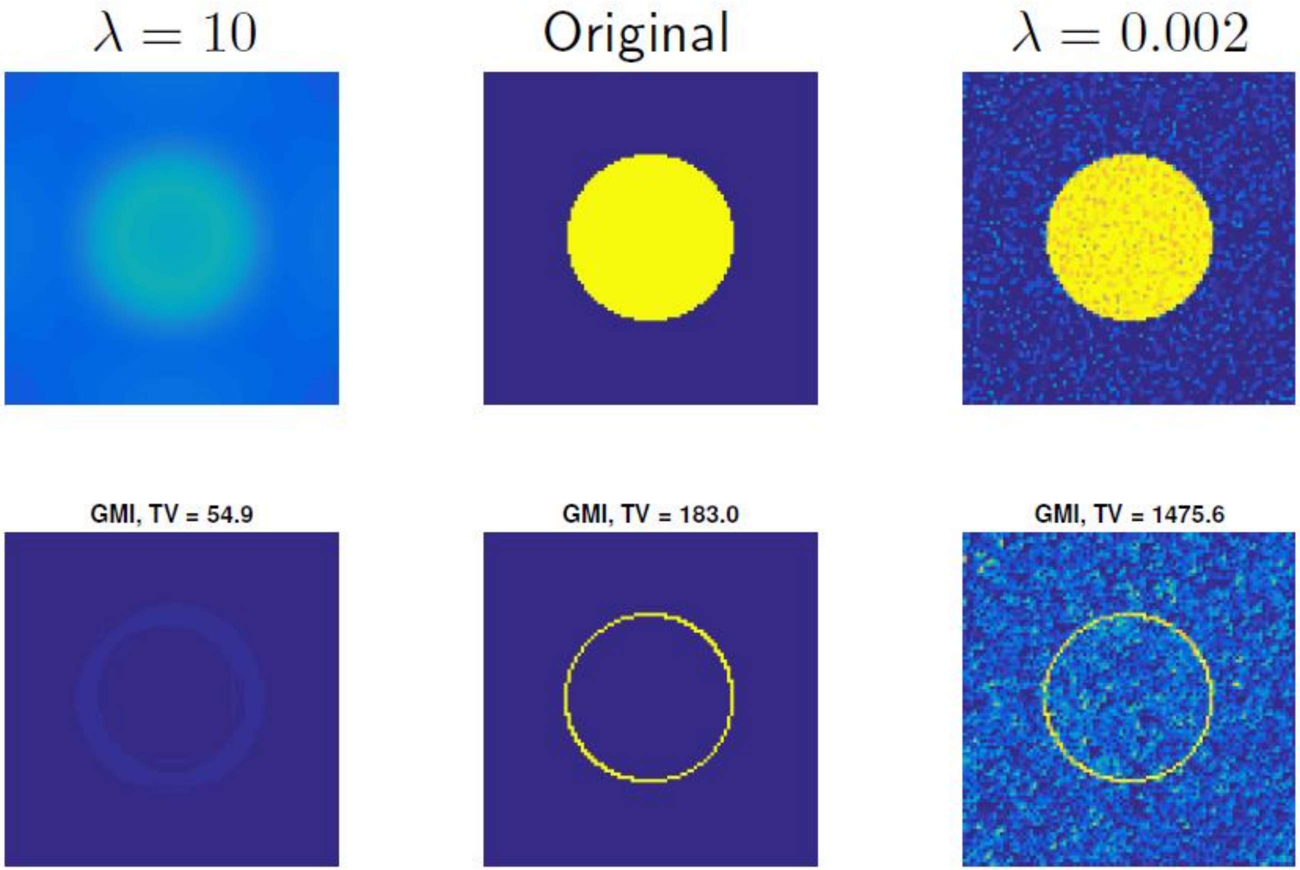
$$\min_{u} \; \|Au - b\|_2^2 + \lambda \cdot \mathrm{TV}(u)$$



- Large λ: Almost only effect of regularizer. TV $\rightarrow$ Constant.
- Small λ: Almost just least-squares solution.
- Best trade-off?

What is TV?

- Measures variation of an image.
- Sum gradient magnitude image (GMI): $\mathrm{TV}(u) = \sum_j \|D_j u\|_2$



- Prior: Few homogeneous regions with simple boundaries.
- Quite succesful in tomography, in particular for reduced data.

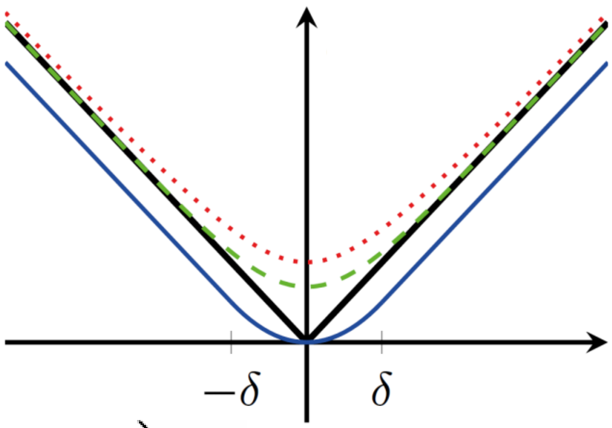
How to solve TV optimisation problem?

- TV is NOT smooth, i.e., NOT differentiable – due to coupling of *x* and *y* derivatives under a square-root:

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

- We cannot use gradient descent etc.

- One approach is to *smooth the problem:*



$$\mathrm{TV}_\delta(u) = \sum_{i,j} \left(\sqrt{(D_y u)^2 + (D_x u)^2 + \delta^2} \right)_{i,j}$$

- Only an approximation – smoothing effects may occur.
- Additional parameter δ and speed vs accuracy trade-off.

FISTA: Fast Iterative Shrinkage Thresholding Algorithm

- Optimisation problem: $$\boldsymbol{x}^\star = \arg\min_{\boldsymbol{x}} \{\mathcal{F}(\boldsymbol{x}) + \beta\, \mathcal{G}(\boldsymbol{x})\}$$
- $\mathcal{F}$ is a smooth (differentiable) function and $\mathcal{G}$ is (possibly) non-smooth.
- **FISTA pseudo code:**

Lipschitz constant of $\mathcal{F}$

Input: $\boldsymbol{b}$, $\boldsymbol{x}^{[0]}$, β, S, L

Number of iterations to run

Output: $\boldsymbol{x}^{[S]}$

$\boldsymbol{y}^{[1]} = \boldsymbol{x}^{[0]}$, $t^{[1]} = 1$

for all $s = 1, \ldots, S$ **do**

1: $\boldsymbol{u}^{[s]} = \boldsymbol{y}^{[s]} - L^{-1}\nabla\mathcal{F}(\boldsymbol{y}^{[s]})$ → **Gradient step**

2: $\boldsymbol{x}^{[s]} = \mathrm{prox}_{\beta/L}[\mathcal{G}](\boldsymbol{u}^{[s]})$ → **Proximal mapping**

3: $t^{[s+1]} = \left(1 + \sqrt{1 + 4(t^{[s]})^2}\right)/2$

4: $\boldsymbol{y}^{[s+1]} = \boldsymbol{x}^{[s]} + (t^{[s]} - 1)/t^{[s+1]} \cdot (\boldsymbol{x}^{[s]} - \boldsymbol{x}^{[s-1]})$

end for

Beck and Teboulle (2009), https://doi.org/10.1137/080716542

Proximal mapping

- Defined through a minimisation problem

$$\mathrm{prox}_{\beta/L}[\mathcal{G}](\boldsymbol{v}) = \arg\min_{\boldsymbol{u}} \left\{ \frac{\beta}{L}\mathcal{G}(\boldsymbol{u}) + \frac{1}{2}\|\boldsymbol{u}-\boldsymbol{v}\|_2^2 \right\}$$

- FISTA is useful when proximal mapping above has simple closed-form solution or can be efficiently computed numerically.

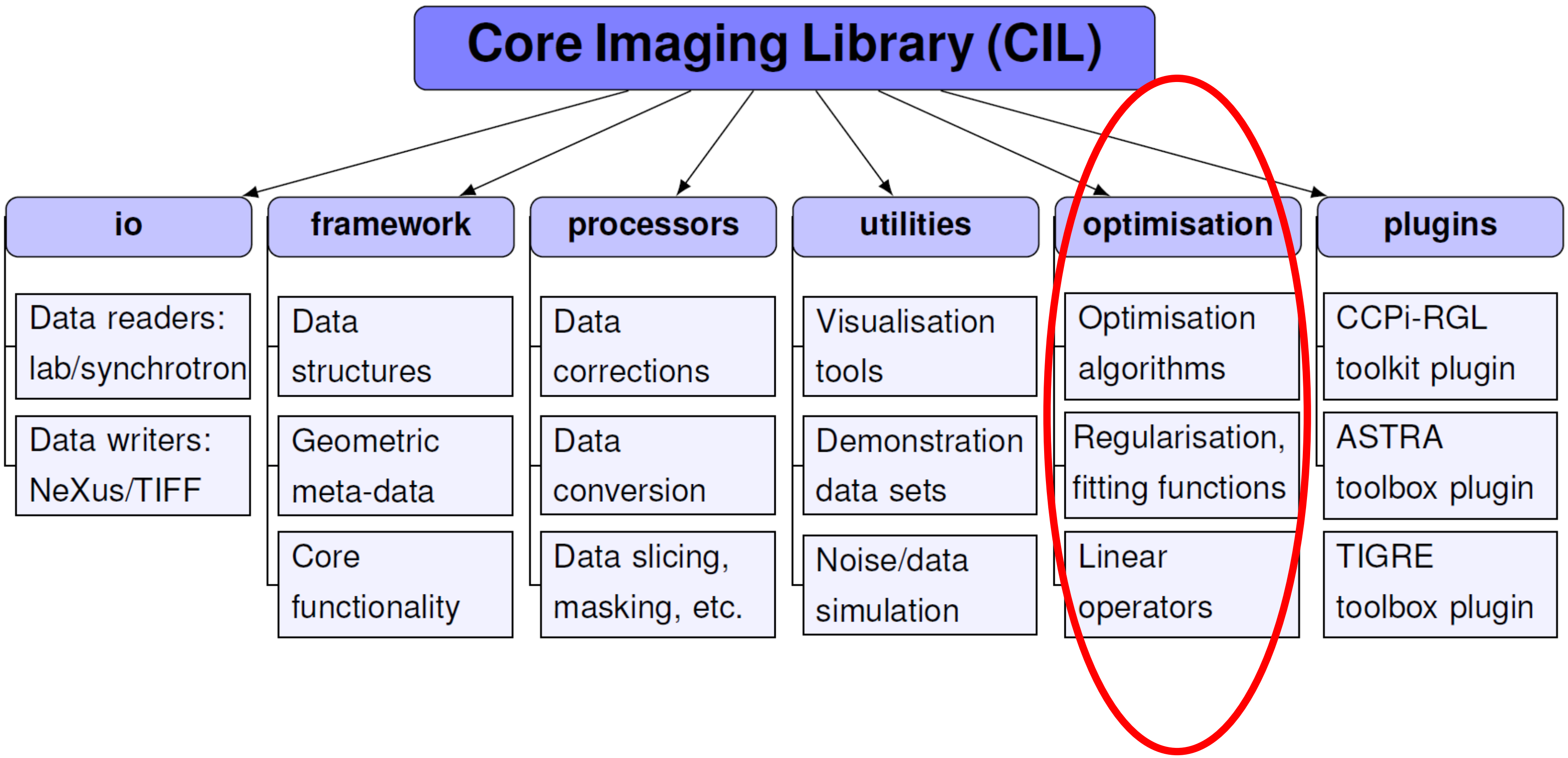
- Simple closed-form examples for $\mathcal{G}$:
  - Constraint to convex set: Proximal mapping is projection.
  - L1-norm: Proximal mapping is soft-thresholding.

- Proximal mapping for TV can be computed numerically.

Parikh and Boyd, Proximal Algorithms (2014), http://dx.doi.org/10.1561/2400000003

Optimisation-based reconstruction in CIL

CIL module structure and contents



The **cil.plugins** module contains wrapper code for other software and third-party libraries that need to be installed separately to be used by CIL.

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

Operators in CIL

name	description
BlockOperator	form block (array) operator from multiple operators
BlurringOperator	apply point spread function to blur an image
ChannelwiseOperator	apply the same operator to all channels
DiagonalOperator	form a diagonal operator from image/acquisition data
FiniteDifferenceOperator	apply finite differences in selected dimension
GradientOperator	apply finite difference to multiple/all dimensions
IdentityOperator	apply identity operator, i.e. return input
MaskOperator	from binary input, keep selected entries, mask out rest
SymmetrisedGradientOperator	apply symmetrized gradient, used in TGV
ZeroOperator	operator of all zeroes
ProjectionOperator	tomography forward/back-projection from ASTRA
ProjectionOperator	tomography forward/back-projection from TIGRE

Functions in CIL

name	description
BlockFunction	separable sum of multiple functions
ConstantFunction	function taking the constant value
OperatorCompositionFunction	compose function f and operator A: $f(Ax)$
IndicatorBox	indicator function for box (lower/upper) constraints
KullbackLeibler	Kullback–Leibler divergence data fidelity
L1Norm	L^1-norm: $\|x\|_1 = \sum_i \|x_i\|$
L2NormSquared	squared L^2-norm: $\|x\|_2^2 = \sum_i x_i^2$
LeastSquares	least-squares data fidelity: $\|Ax - b\|_2^2$
MixedL21Norm	mixed $L^{2,1}$-norm: $\|(U_1; U_2)\|_{2,1} = \|(U_1^2 + U_2^2)^{1/2}\|_1$
SmoothMixedL21Norm	smooth $L^{2,1}$-norm: $\|(U_1; U_2)\|_{2,1}^S = \|(U_1^2 + U_2^2 + \beta^2)^{1/2}\|_1$
WeightedL2NormSquared	weighted squared L^2-norm: $\|x\|_w^2 = \sum_i (w_i \cdot x_i^2)$
TotalVariation	$\mathrm{TV}(u) = \|Du\|_{2,1} = \sum_{i,j} \left(\sqrt{(D_y u)^2 + (D_x u)^2}\right)_{i,j}$

Algorithms in CIL

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

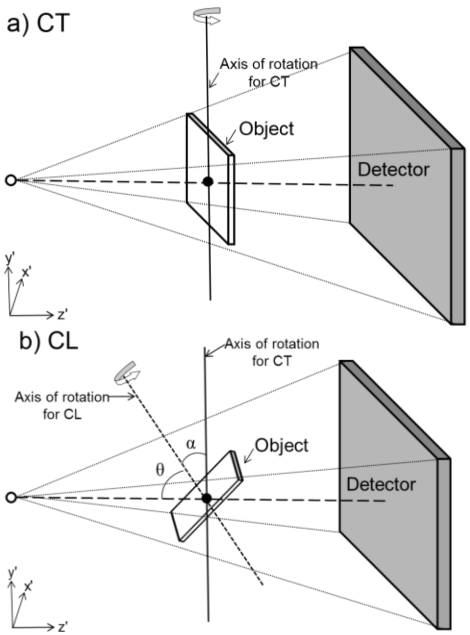
Example: Total Variation in CIL

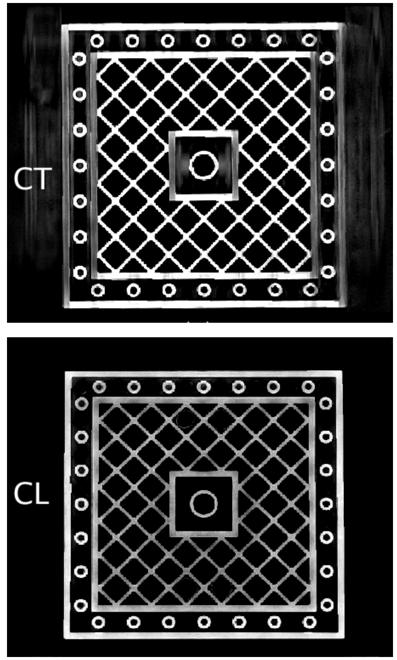
```
F = LeastSquares(A, b)

G = alpha*TotalVariation()

algo = FISTA(f=F,
             g=G,
             initial=x0,
             max_iteration=1000)

algo.run(50, verbose=1)

show2D(algo.solution)
```

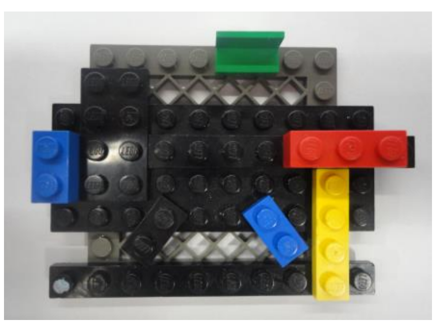
See CIL notebook:
2_Iterative/01_optimisation_gd_fista.ipynb

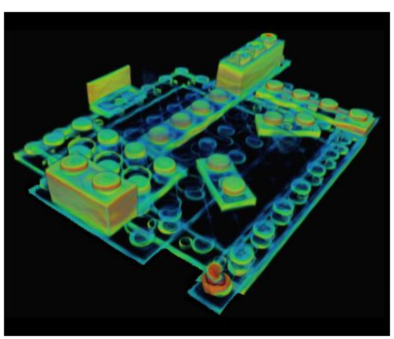
Example cases using CIL

Tomographic imaging of planar samples with laminography

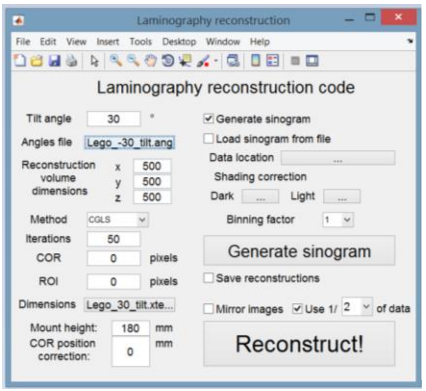


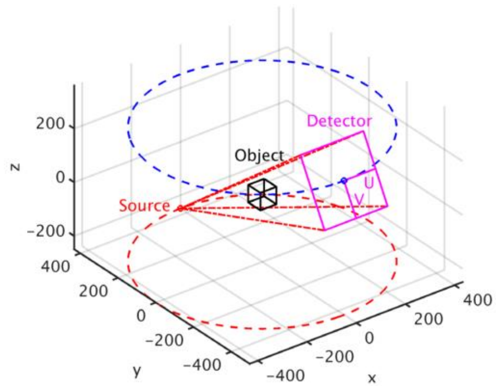






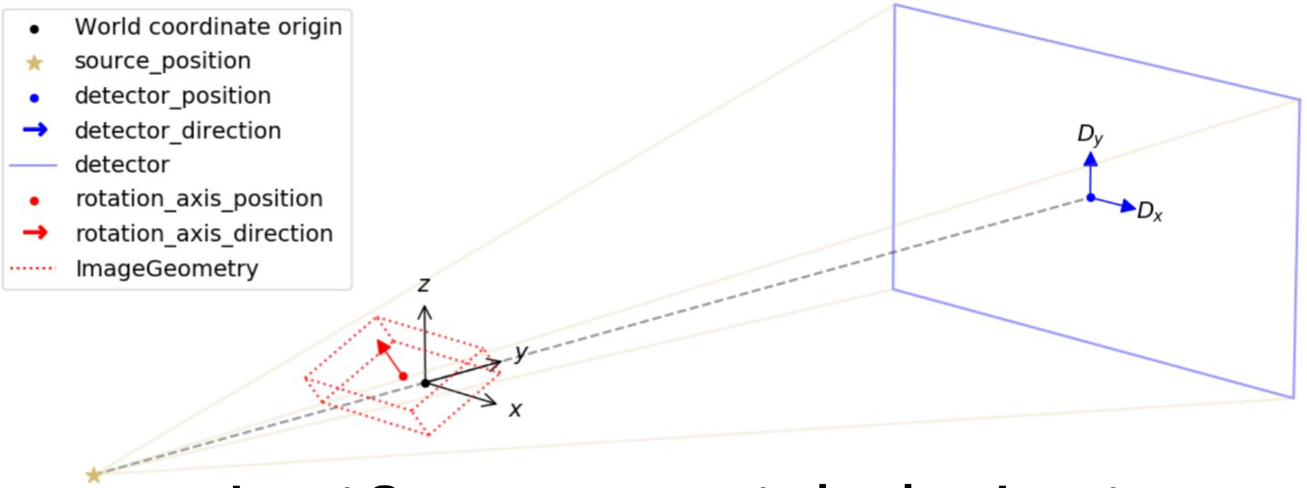
- Planar samples like composite panels and printed circuit boards difficult to scan due to different exposure along views.
- Conventional scan gives limited-angle artifacts, missing edges.
- Laminography allows uniform exposure.
- Non-standard geometry needs dedicated reconstruction – here used CGLS.

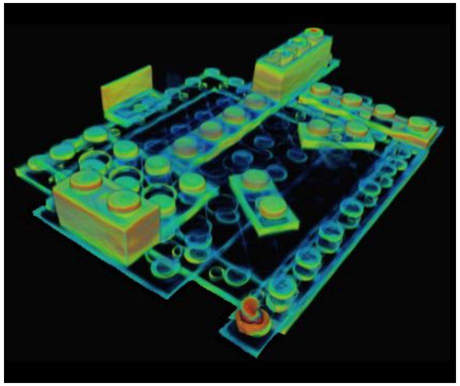




Fisher, Holmes, Jørgensen, Gajjar, Behnsen, Lionheart & Withers, Meas. Sci. Technol. 30 (2019), pp. 035401

Laminography artifacts reduced by TV

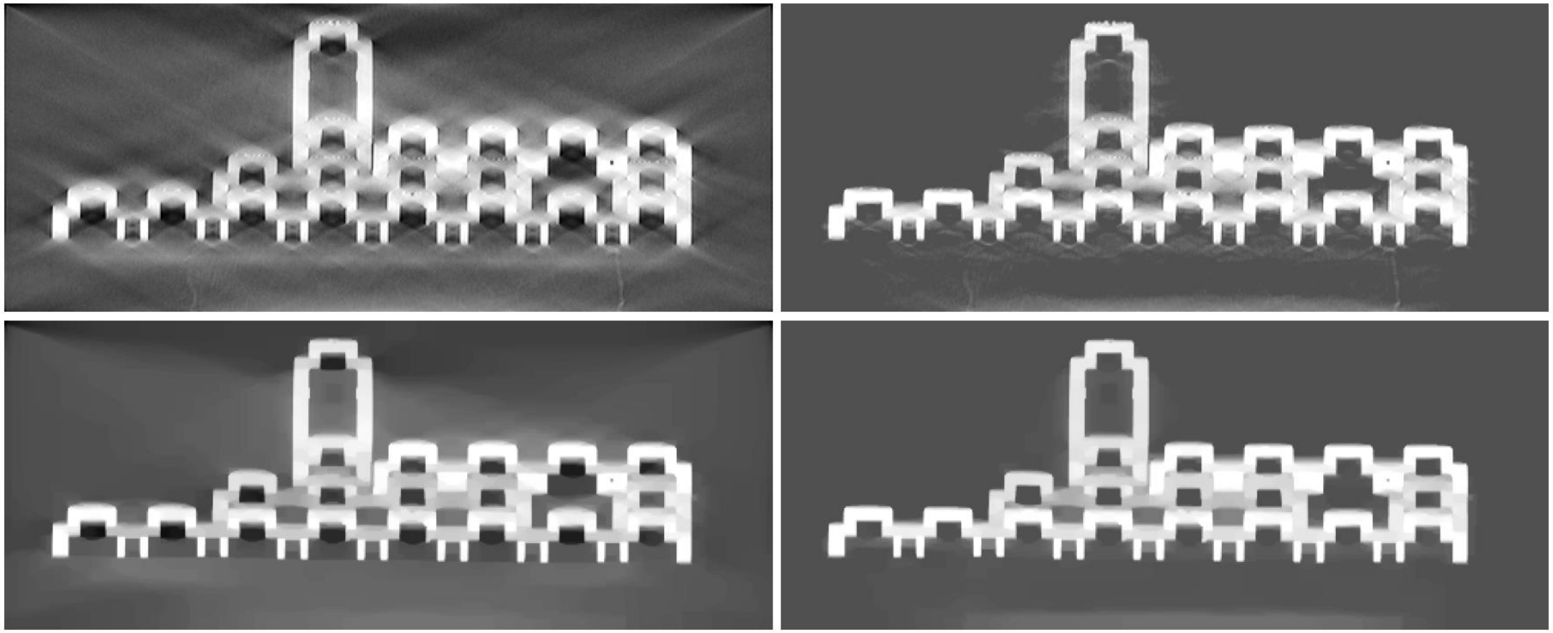




Least Squares, unconstrained | Least squares, nonnegativity

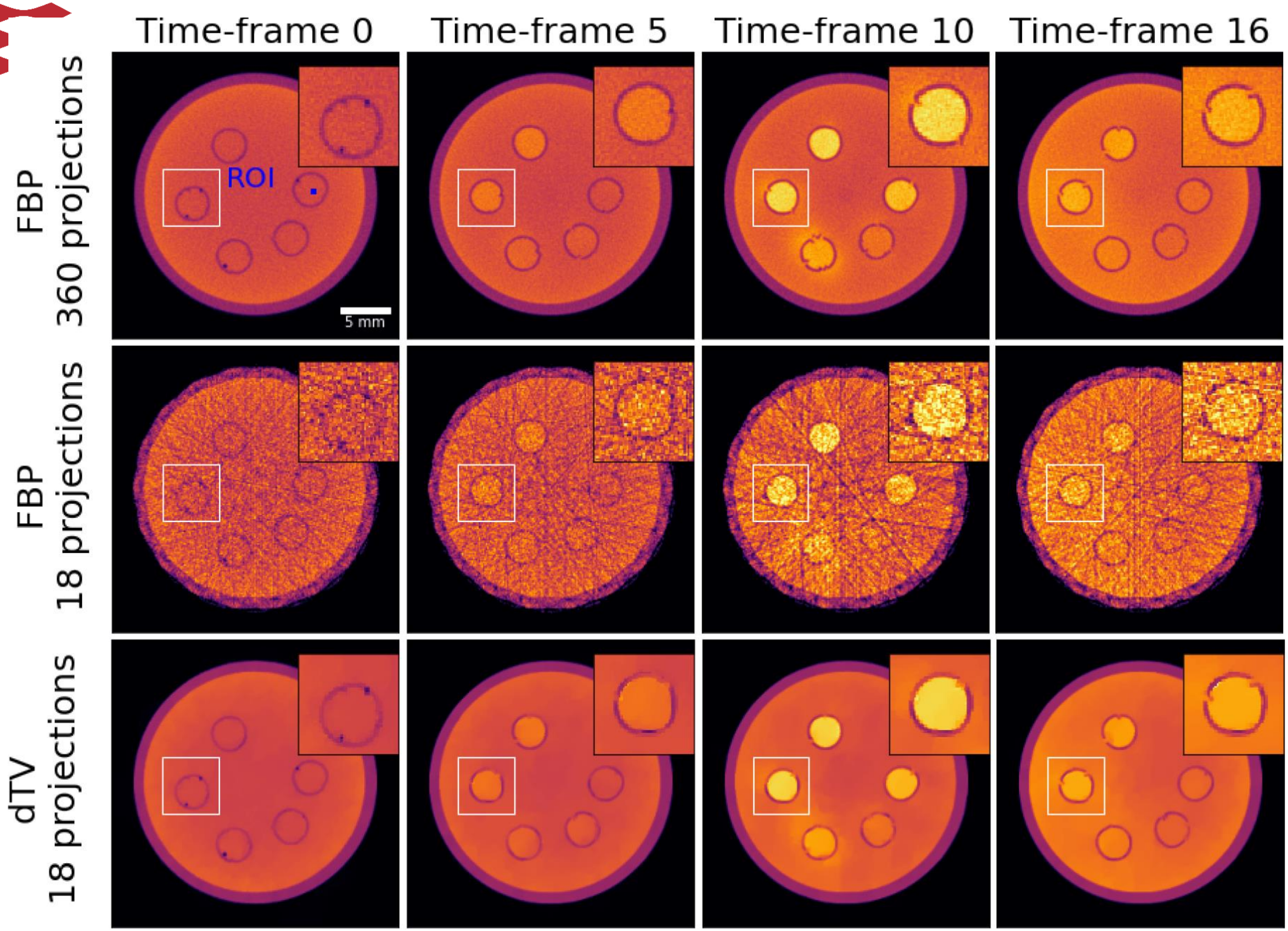
No regu.

TV regu.



J. et al. 2021: *Core Imaging Library - Part I: a versatile Python framework for tomographic imaging,* Phil Trans A. https://doi.org/10.1098/rsta.2020.0192

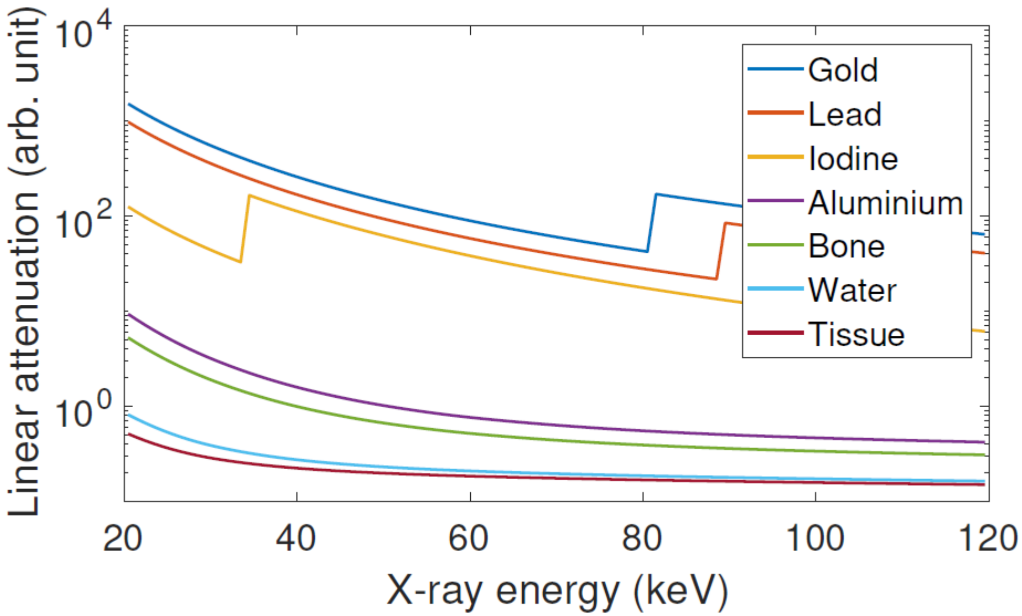
Dynamic CT with directional TV using reference images



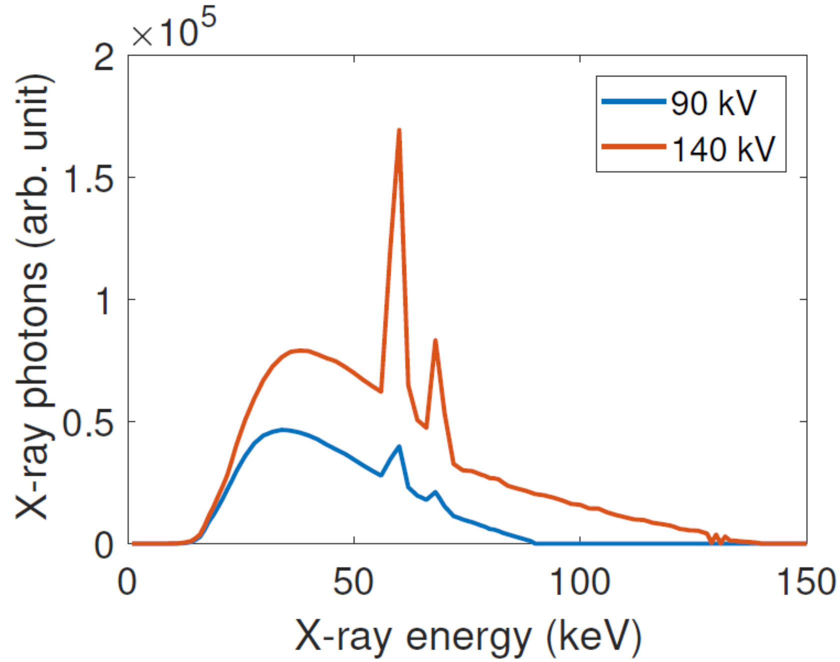
Papoutsellis et al. 2021: *Core Imaging Library - Part II: multichannel reconstruction for dynamic and spectral tomography*, Phil Trans A. https://doi.org/10.1098/rsta.2020.0193

X-ray beam normally not mono-chromatic

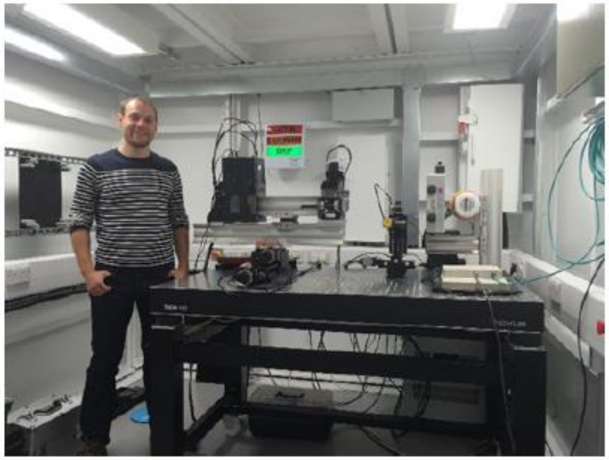
Attenuation is energy-dependent



X-ray source spectrum

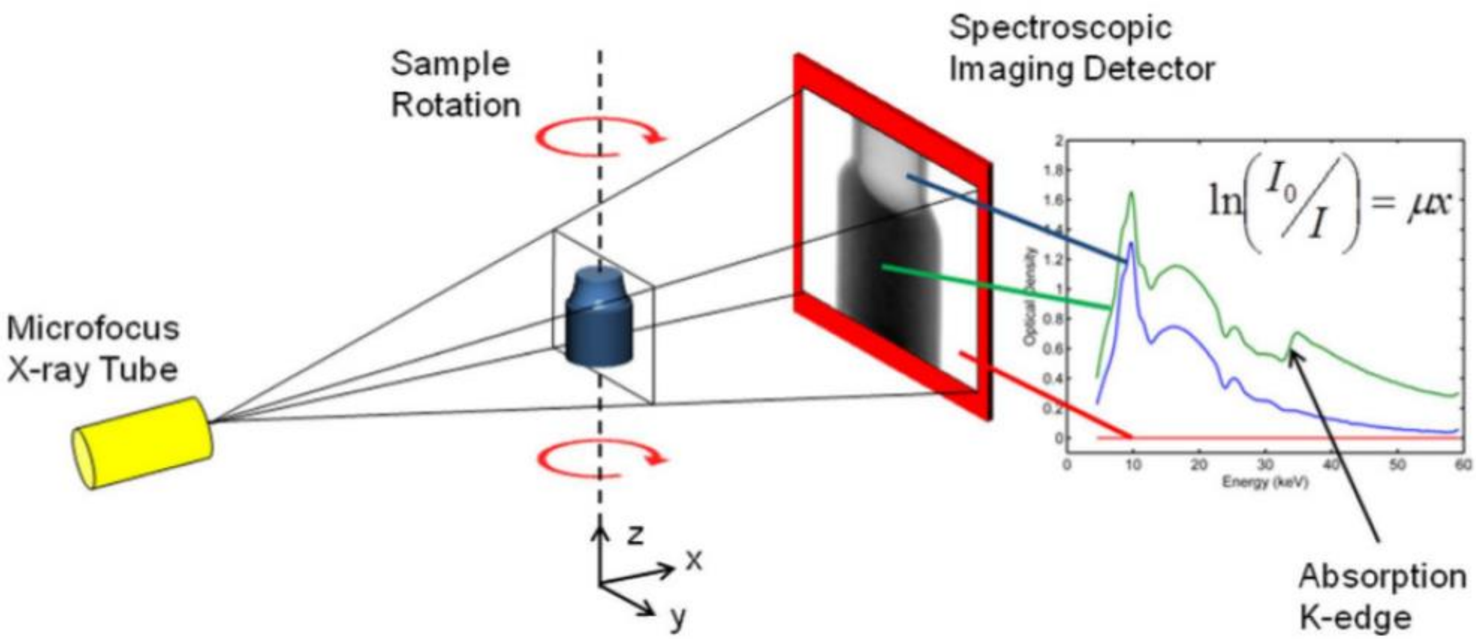


Hyperspectral X-ray CT @ Manchester



The Manchester Colour Bay instrument for hyperspectral X-ray imaging using a HEXITEC detector

(C. Egan et al. 3D chemical imaging in the laboratory by hyperspectral X-ray CT, 2015).



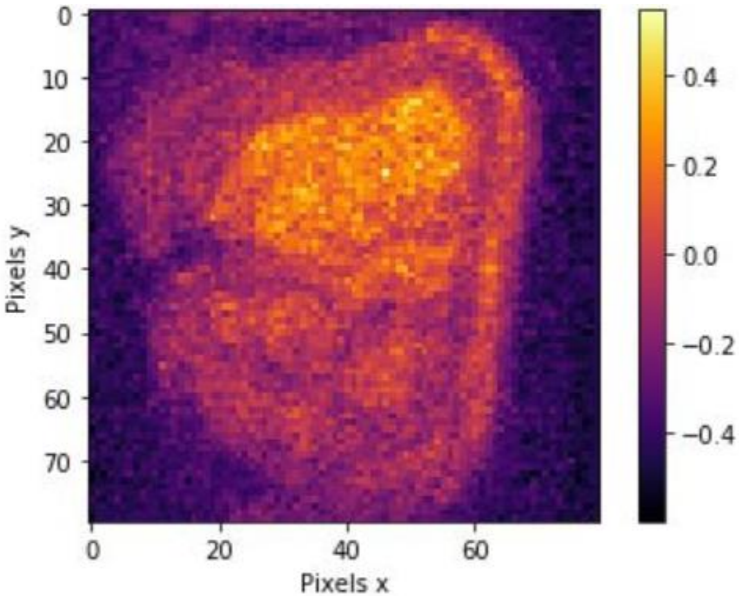
Why new reconstruction methods and software needed?

- Commerical software, e.g., Nikon CT Pro, Octopus
- Limited possibilities, mostly just FBP/FDK
- Some open source reconstruction tools: ASTRA, tomopy, Savu, ...
- Mostly parallel-beam, no software for multichannel CT

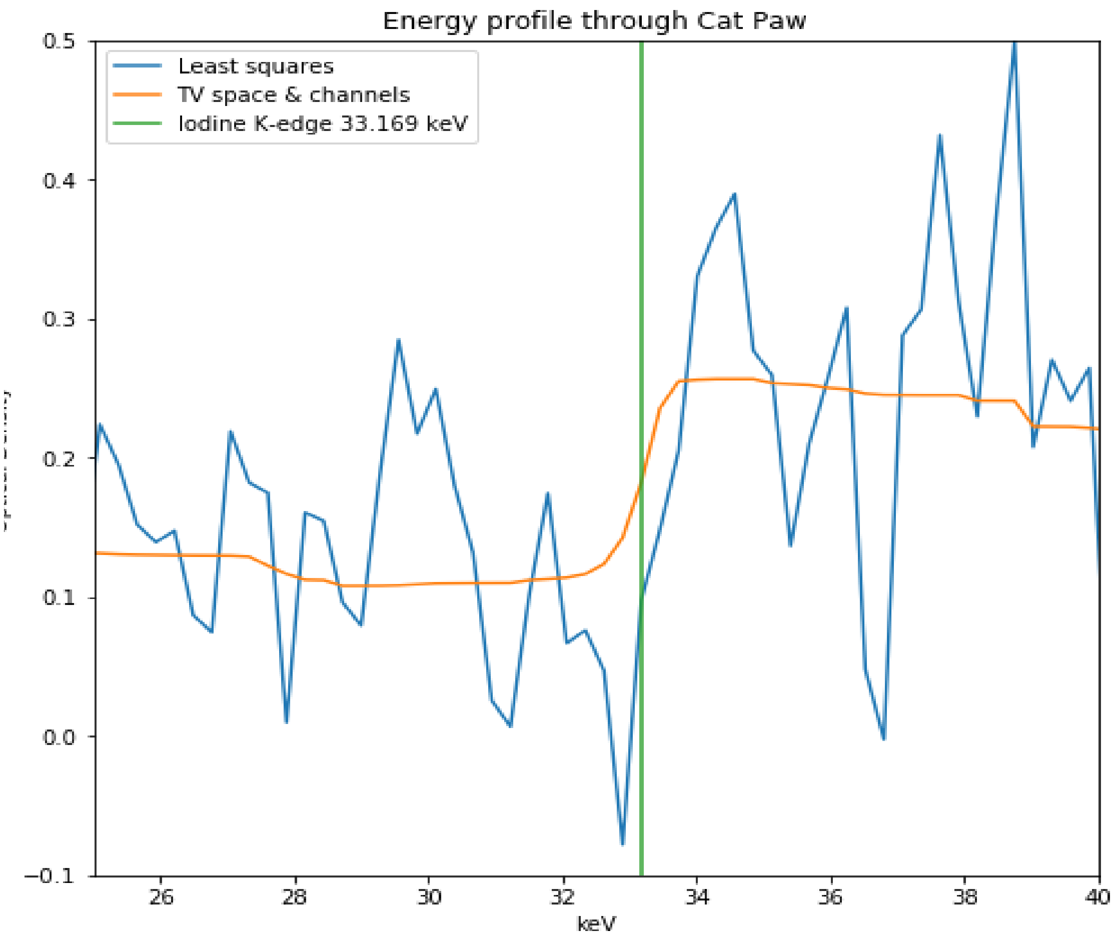
Challenges and opportunities:

- Few counts in each channel
- Naïve channel-wise reconstruction poor quality
- Neighbouring channels mostly similar
- Explore how to regularise across channels (and space) to improve reconstruction quality.

Iodine stained cat paw @ Colour Bay



Single Projection of cat paw, with regions of increased optical density due to iodine

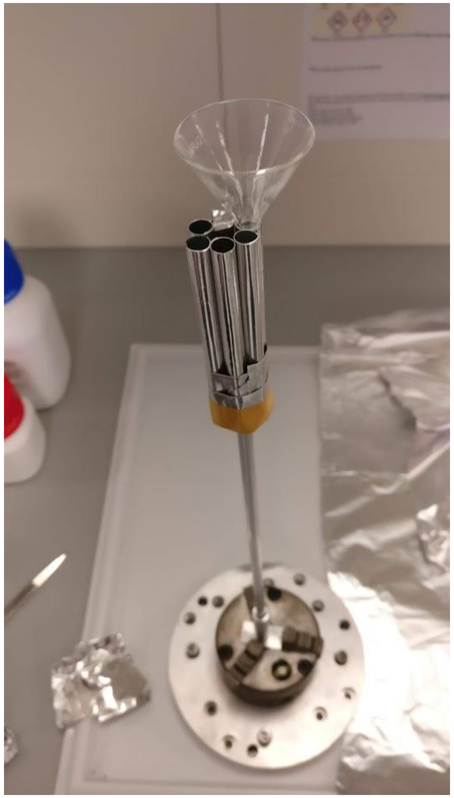
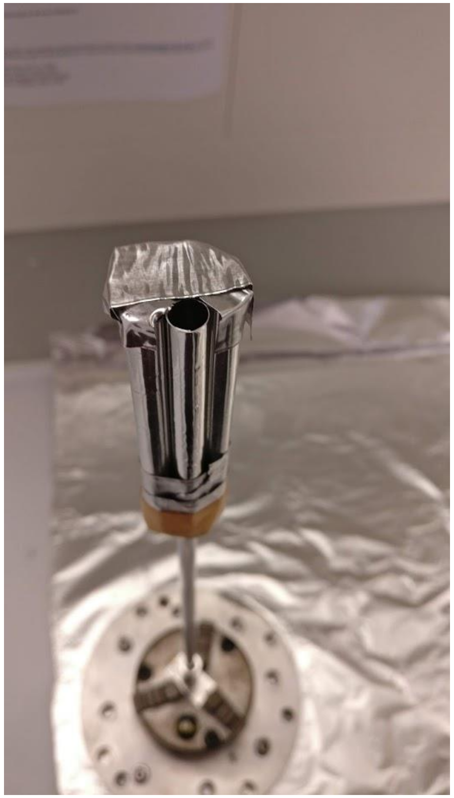


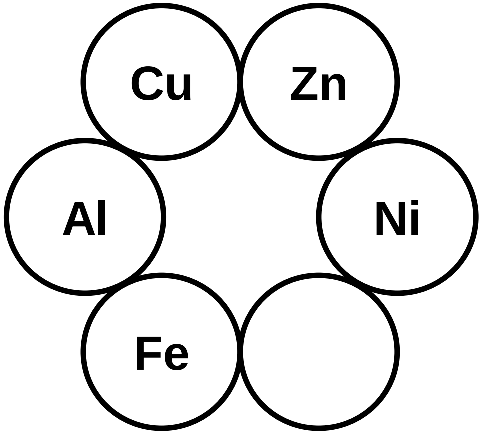
Spectral Profile, CGLS vs. PDHG TV

IMAT hyperspectral neutron data

5 aluminum foil cylinders
filled with metallic powder
+ 1 empty

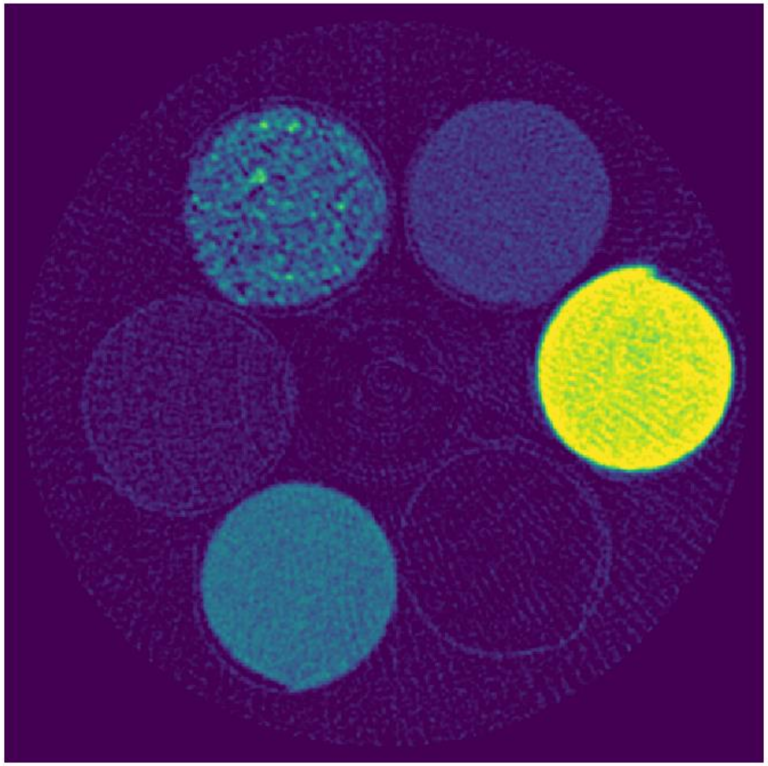
Detector size: 512x512, 0.055 mm pixel
120 projections over 180°
2840 energy channels between 1 and 5 A
15 min exposure time

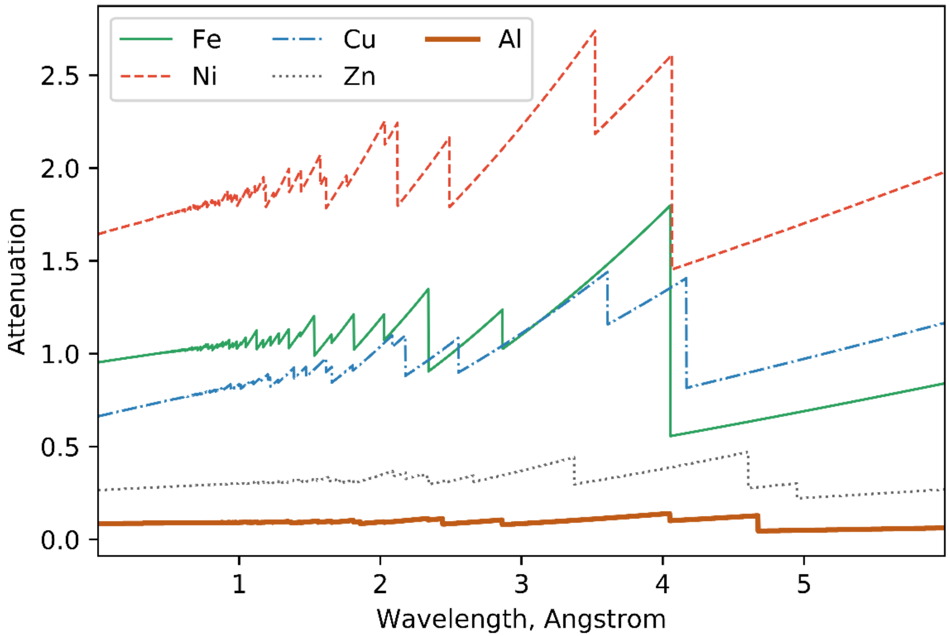


Propose TV spatially and TGV spectrally

Spatially:
Piecewise **constant** plus jumps

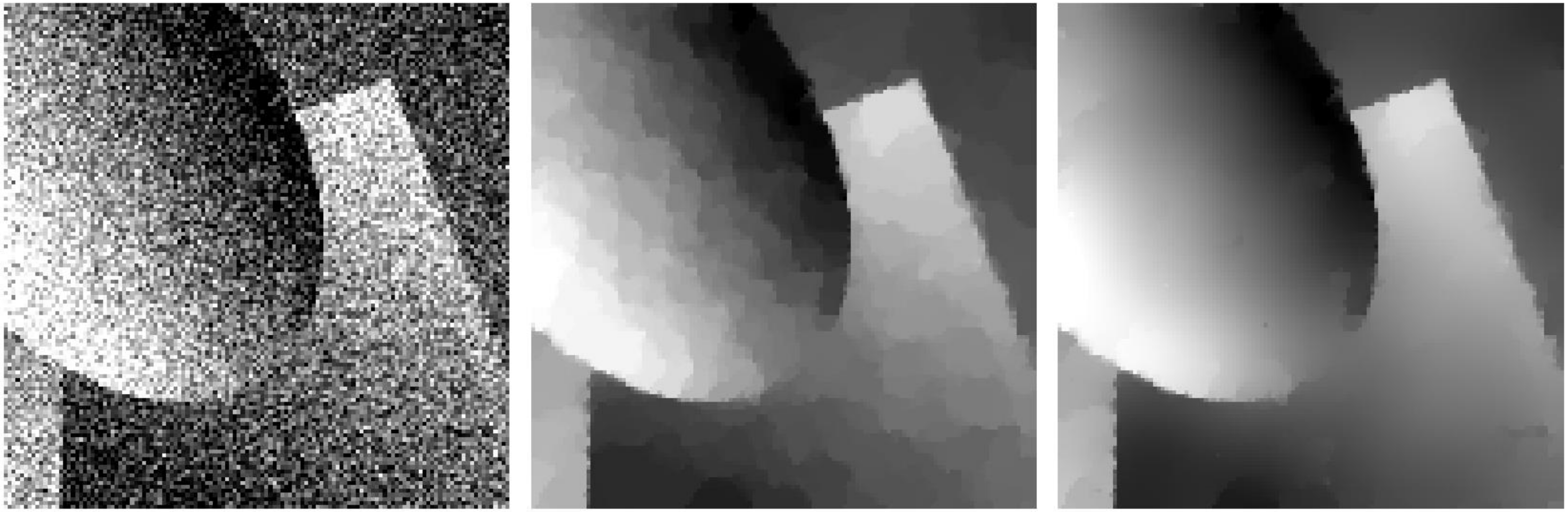


Spectrally:
Piecewise **smooth** plus jumps



Total Generalised Variation (TGV)

- TV: For piecewise constant plus jumps.
- TGV: For piecewise smooth plus jumps.



noisy image TV denoising TGV denoising

DTU

Spatial TV plus spectral TGV optimisation problem

$$\arg\min_{u} \ \alpha\,\mathrm{TV}(u_{space}) + \beta\,\mathrm{TGV}(u_{spt}) + \frac{1}{2}\|g-u\|_2^2$$

$$\arg\min_{u,w} \ \alpha\,\|\nabla u\|_{2,1} + \beta\left(\|\partial_{spt}u - w\|_1 + \sqrt{2}\|\partial_{spt}w\|_1\right) + \frac{1}{2}\|g-u\|_2^2$$

$$\Downarrow$$

$$\arg\min_{x} \ \mathcal{F}(K\,x) + \mathcal{G}(x)$$

$$x = \begin{bmatrix} u \\ w \end{bmatrix} \qquad \mathcal{G}(x) = \mathbb{I}_{\{u>0\}}(u) \qquad K = \begin{bmatrix} \nabla & \mathbb{O} \\ \partial_{spt} & -\mathbb{I} \\ \mathbb{O} & \partial_{spt} \\ \mathbb{I} & \mathbb{O} \end{bmatrix}$$

$$\begin{aligned}\mathcal{F}(z_1,z_2,z_3,z_4) &= F_1(z_1)+F_2(z_2)+F_3(z_3)+F_4(z_4)\\ &= \alpha\,\|z_1\|_{2,1} + \beta\,\|z_2\|_1 + \beta\sqrt{2}\|z_3\|_1 + \frac{1}{2}\|g-z_4\|_2^2\end{aligned}$$

Spatial TV plus spectral TGV implementation in CIL

$$\arg\min_{x} \; \mathcal{F}(K\,x) + \mathcal{G}(x)$$

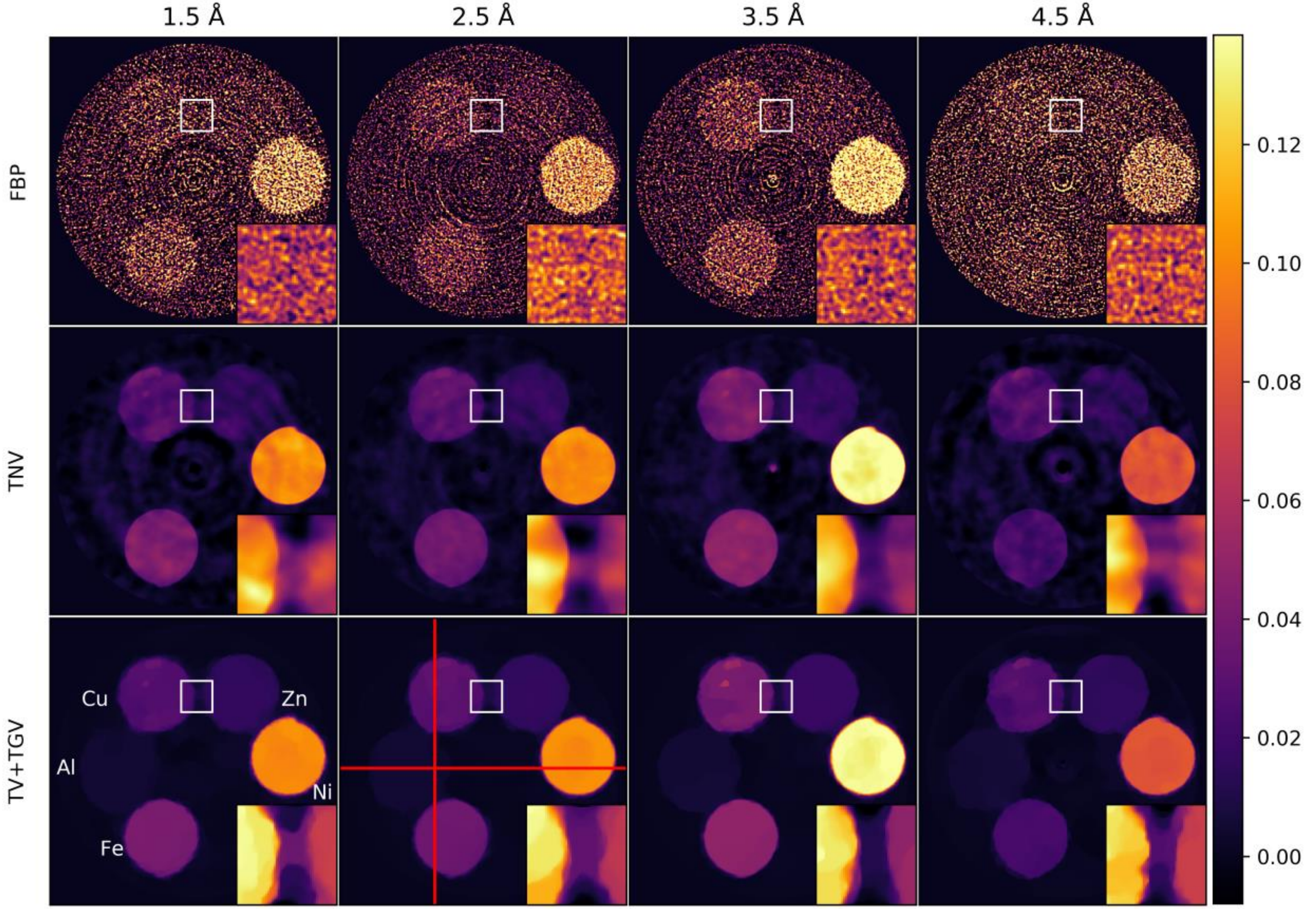
```
# Define Operator K
op11 = Gradient(ig, correlation='Space')
op12 = ZeroOperator(ig, op11.range_geometry())

op21 = FiniteDiff(ig, direction = 0)
op22 = -Identity(ig)

op31 = ZeroOperator(ig)
op32 = FiniteDiff(ig, direction = 0)

op41 = Identity(ig)
op42 = ZeroOperator(ig)

OP = BlockOperator(op11, op12,
                   op21, op22,
                   op31, op32,
                   op41, op42,shape=(4,2) )
```

```
# Define Function G, with positivity constraint
G = Indicator(lower=0)
```

```
# Define Separable Function F
alpha = 0.3
beta = 0.02
gamma = np.sqrt(2) * beta
f1 = alpha * MixedL21Norm()
f2 = beta * L1Norm()
f3 = gamma * L1Norm()
f4 = 0.5 * L2NormSquared(b=g)
F = BlockFunction(f1, f2, f3, f4)
```

```
# Compute operator Norm
normK = operator.norm()

# Primal & dual stepsizes
sigma = 1
tau = 1/(sigma*normK**2)

# Setup and run the PDHG algorithm
pdhg = PDHG(f=F,g=G,operator=operator,
tau=tau, sigma=sigma)
pdhg.run(200)
```

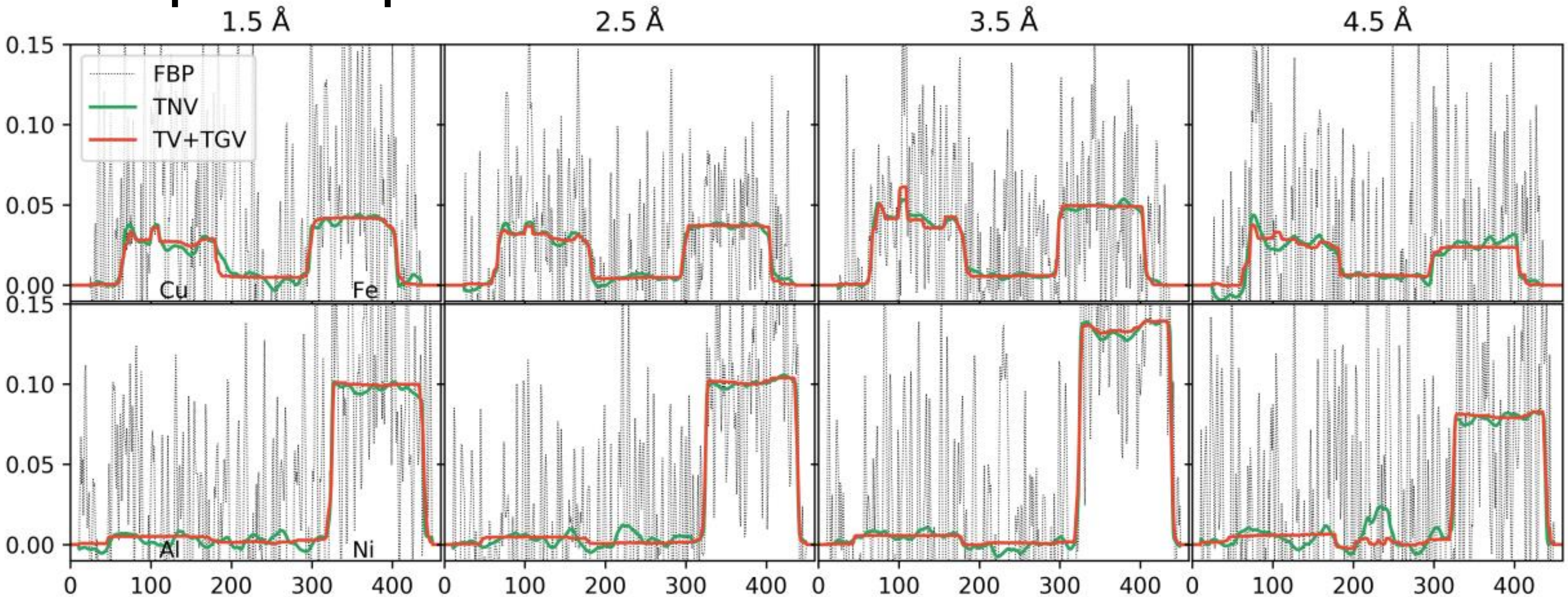
Reconstructed channel images



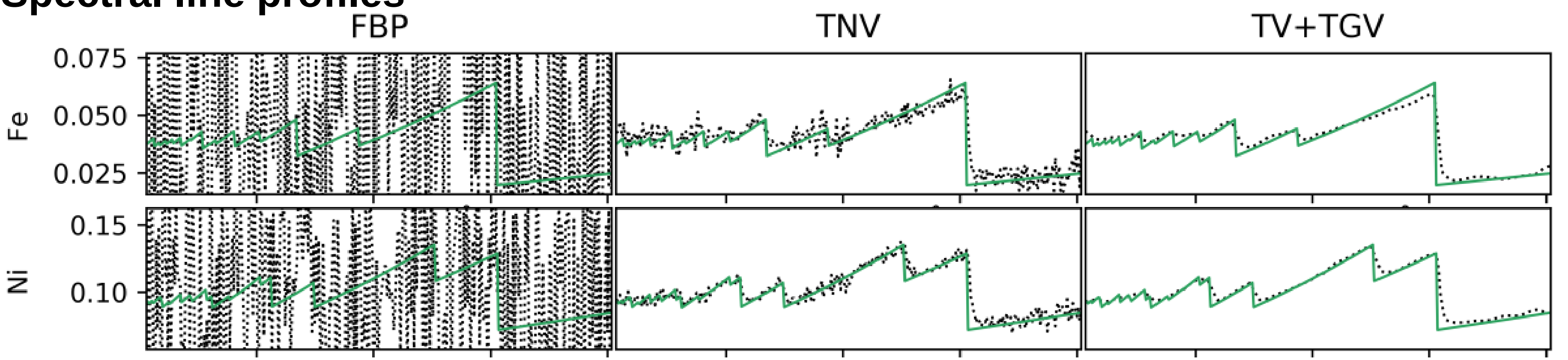
Ametova et al. 2021: *Crystalline phase discriminating neutron tomography using advanced reconstruction methods*, J. Physics D, https://doi.org/10.1088/1361-6463/ac02f9

Hyperspectral neutron CT with spatio-spectral regu.

Spatial line profiles



Spectral line profiles



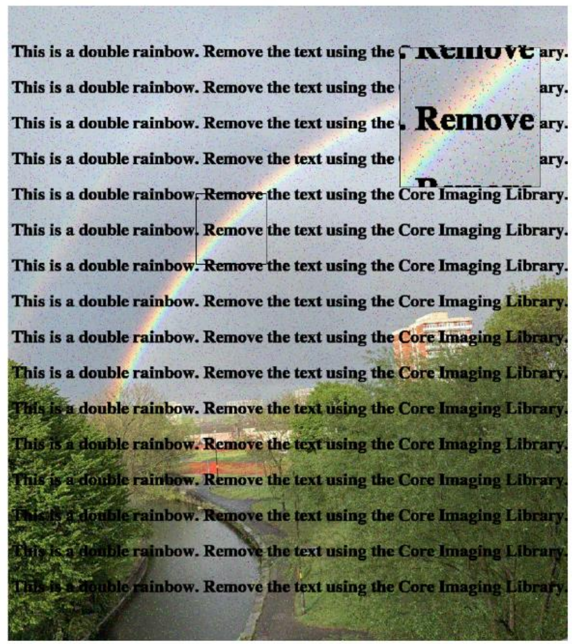
Ametova et al. 2021: *Crystalline phase discriminating neutron tomography using advanced reconstruction methods*, J. Physics D, https://doi.org/10.1088/1361-6463/ac02f9

Other (linear) inverse problems in CIL

Colour image inpainting and salt/pepper denoising using L1-norm data fidelity and total generalized variation (TGV)

Ground truth

Corrupted image



L1 + TGV

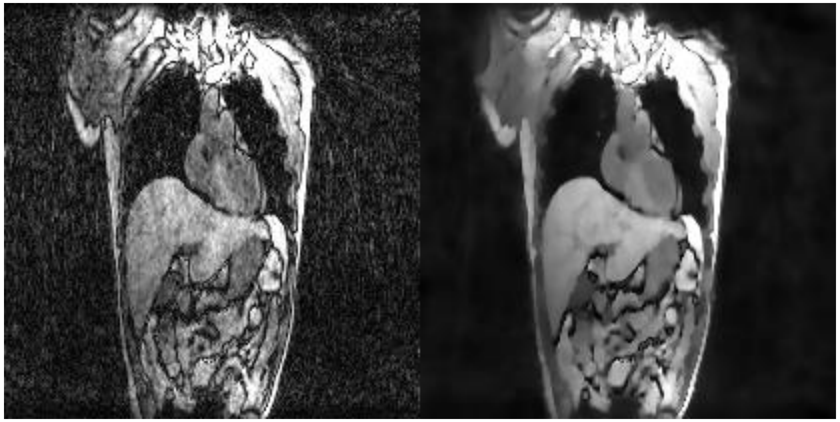
CIL supplies LinearOperators for denoising, deblurring and inpainting problems and users may write a LinearOperator wrapper for their own problem.

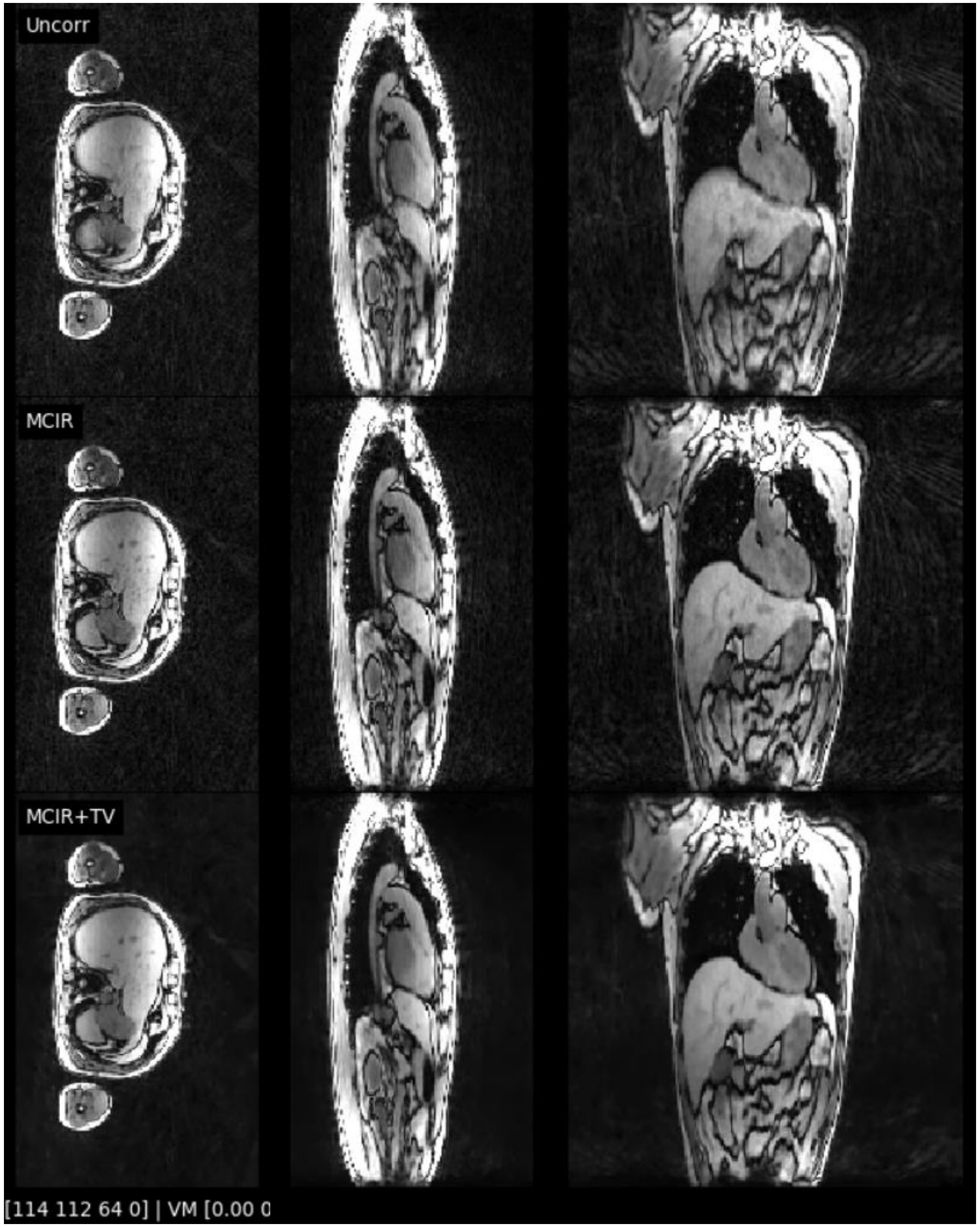
Papoutsellis et al. 2021: *Core Imaging Library - Part II: multichannel reconstruction for dynamic and spectral tomography*, Phil Trans A, https://doi.org/10.1098/rsta.2020.0193

MRI with motion compensation using SIRF and CIL

- SIRF AcquisitionModel for MRI
- CIL reconstruction algorithm

Uncorrected MCIR + TV





Brown et al. 2021: *Motion estimation and correction for simultaneous PET/MR using SIRF and CIL*, Phil Trans A, https://doi.org/10.1098/rsta.2020.0208

Summary

jakj@dtu.dk

Filtered backprojection

- Many projections, full angular range, only moderate noise – look no further!

Iterative reconstruction methods

- Solve optimization problem numerically to find best image
- Trade-off between fitting data and introducing regularity
- Type of regularizer to use depends on image features

Hyperspectral tomography

- 100s or 1000s or highly noisy channels of tomographic data
- Naive channelwise reconstruction insufficient
- Spatial and spectral regularization such as TV/TGV/TNV improve image quality to allow identification of K-edges/Bragg edges on a single voxel level

Core Imaging Library reconstruction framework in Python

- Single and multichannel reconstruction methods
- Flexible: Easy to mix&match to prototype reconstruction algorithms
- www.ccpi.ac.uk/CIL

- If you are interested in doing a project on CT reconstruction, do contact me!

- BSc, MSc or PhD project – or special course.

- jakj@dtu.dk

- Building 303B, office 111