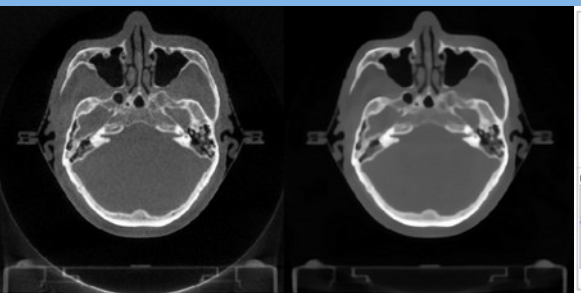
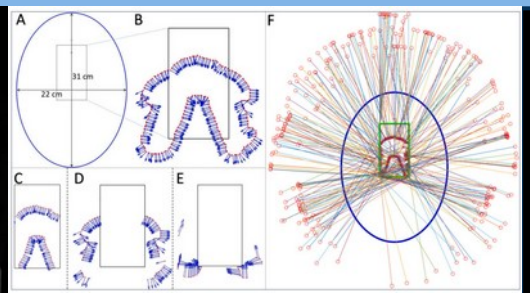
Overview and current challenges of practical CT reconstruction

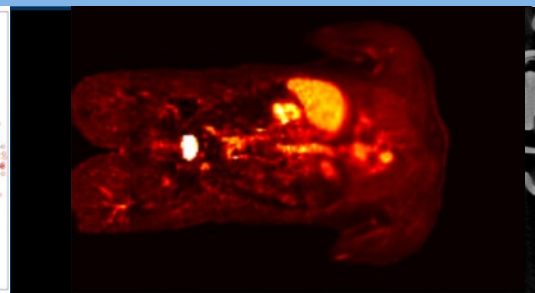


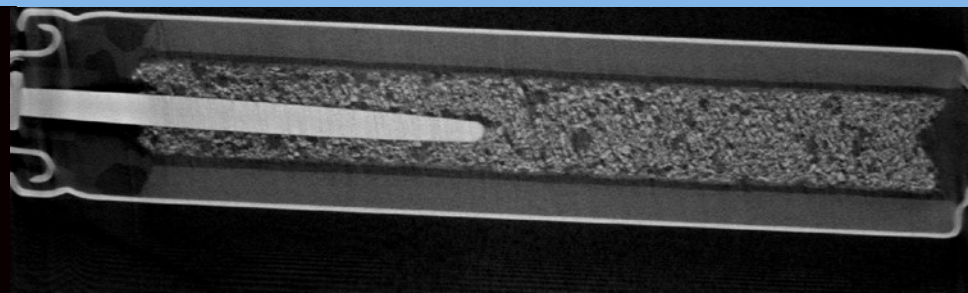
Dr Ander Biguri

Assistant Research Professor working on translating developments in inverse problems (both ML and classical optimization) into applications of tomography in medicine and physics









Index

- Computed Tomography: brief intro
- From maths to hospitals: TIGRE toolbox
- Data driven inverse problems/tomography reconstruction
- What is missing to make the methods applicable to real data?
- My current research
  - LION toolbox
  - Current projects

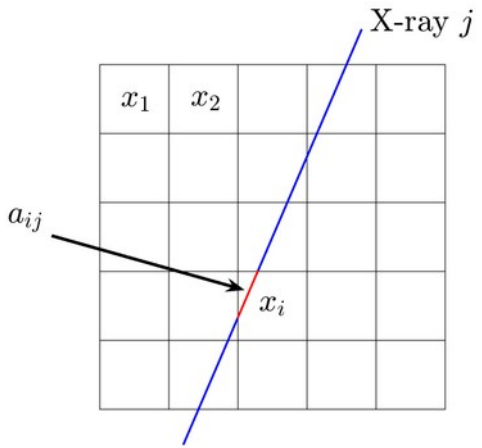
Tomographic Image reconstruction

We are interested in a solution of

$$Ax + e = b, \quad A \in \mathbb{R}^{M \times N}, \quad b \in \mathbb{R}^{M}, \quad e \in \mathbb{R}^{M}$$

coming from suitable discretization of

$$\int_{\Omega} k(s,t) f(t) dt + \varepsilon(s) = g(s).$$



Modeling inverse problems:

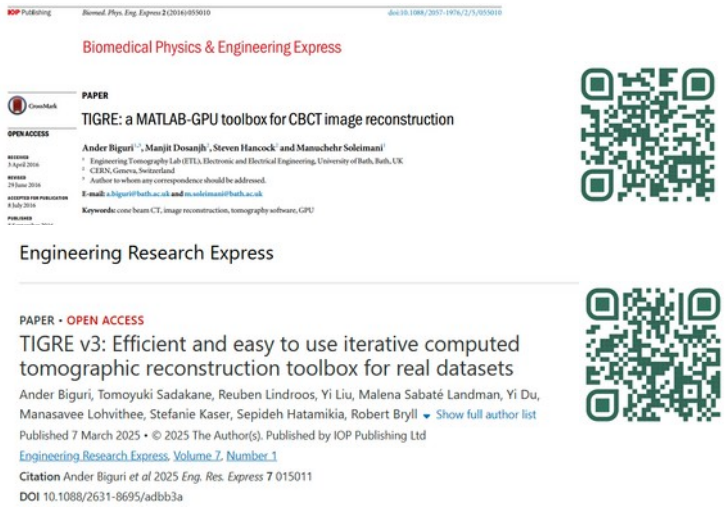
- the process k (resp. A), the output g (resp. b) are known;
- the input f (resp. x) is unknown.

Variational regularization formulation of the problem

- Minimize: $\hat{\mathbf{x}} = \underset{\mathbf{x}}{\operatorname{argmin}} \; \|$
- In 2014: Why do many solvers exist but (almost) none are used in imaging applications?
  - Operator is too computationally expensive
  - Too many scattered algorithm proposals
  - Math is hard! (for imaging scientists)
    - No easy way to try them without investing in maths + software

TIGRE

github.com
/CERN
/TIGRE



- Tomographic Iterative GPU-based reconstruction
- MATLAB/python – CUDA Toolbox
- 3D absorption X ray tomography (Cone & Parallel beam)
- Goals:
  - Modular
  - Easy to use
  - Fast
- Collaborative
  - Open Source
  - Totally Free/ BSD license
  - Encourage contributions

- +500 citations
- Standard tool in academia/industry/healthcare
- +25 reconstruction algorithms from 7 algorithm families
- +7 data loaders from industry/healthcare scans
- +5 businesses use it as backbone of their products

Algorithms (so far)

Analytical methods
- FDK
- FBP

Kaczmarz methods
- SART
- OS-SART
- SIRT

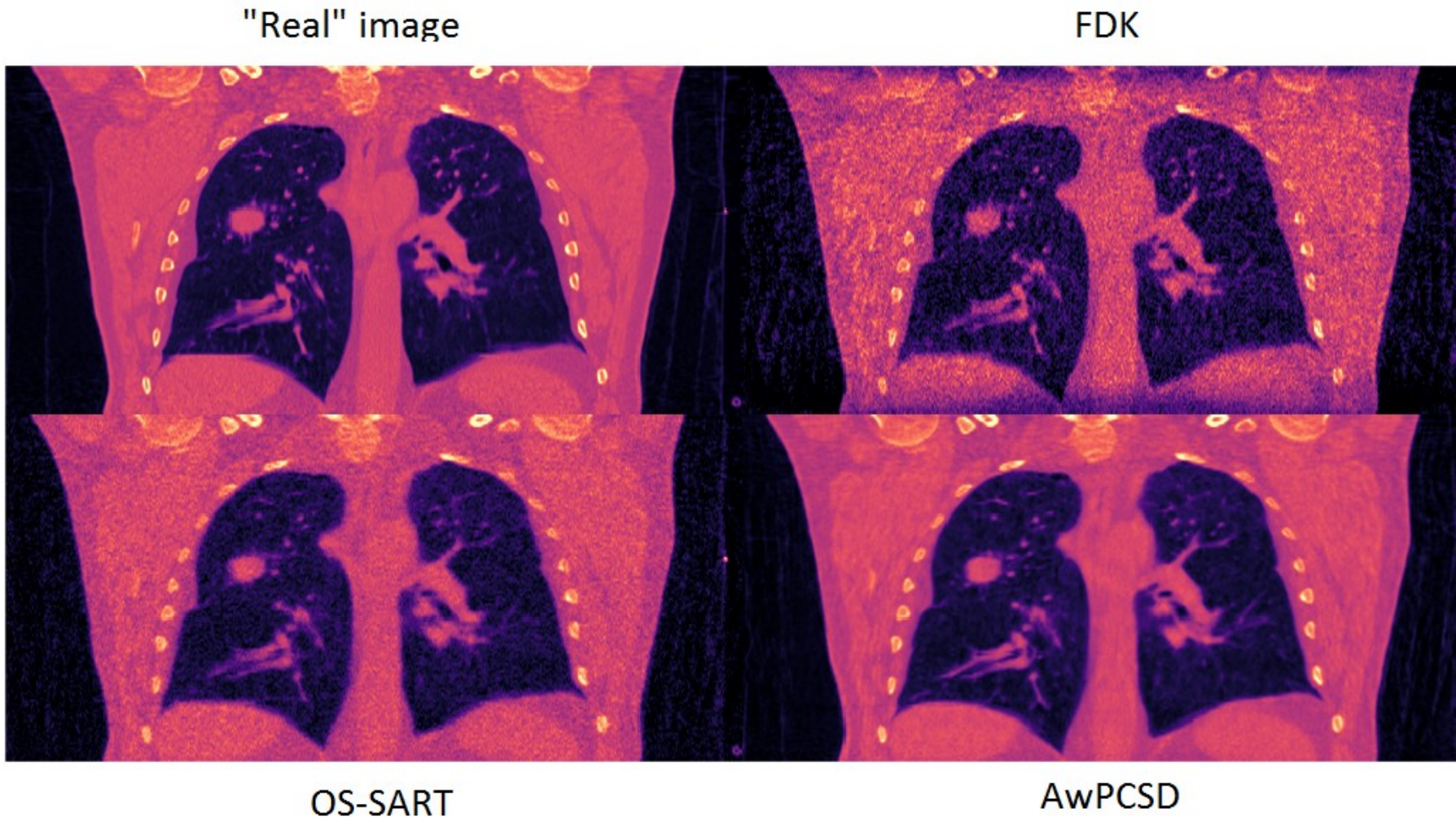
Krylov Subspace
- CGLS
- LSQR
- hLSQR
- LSMR
- AB/BA-GMRES
- CGLS-TV
- h-fLSQR-TV

Statistical methods
- MLEM
- OSEM

POCS methods (with TV gradient reg)
- ASD-POCS
- OS-ASD-POCS
- B-ASD-POCS- β
- SART-TV

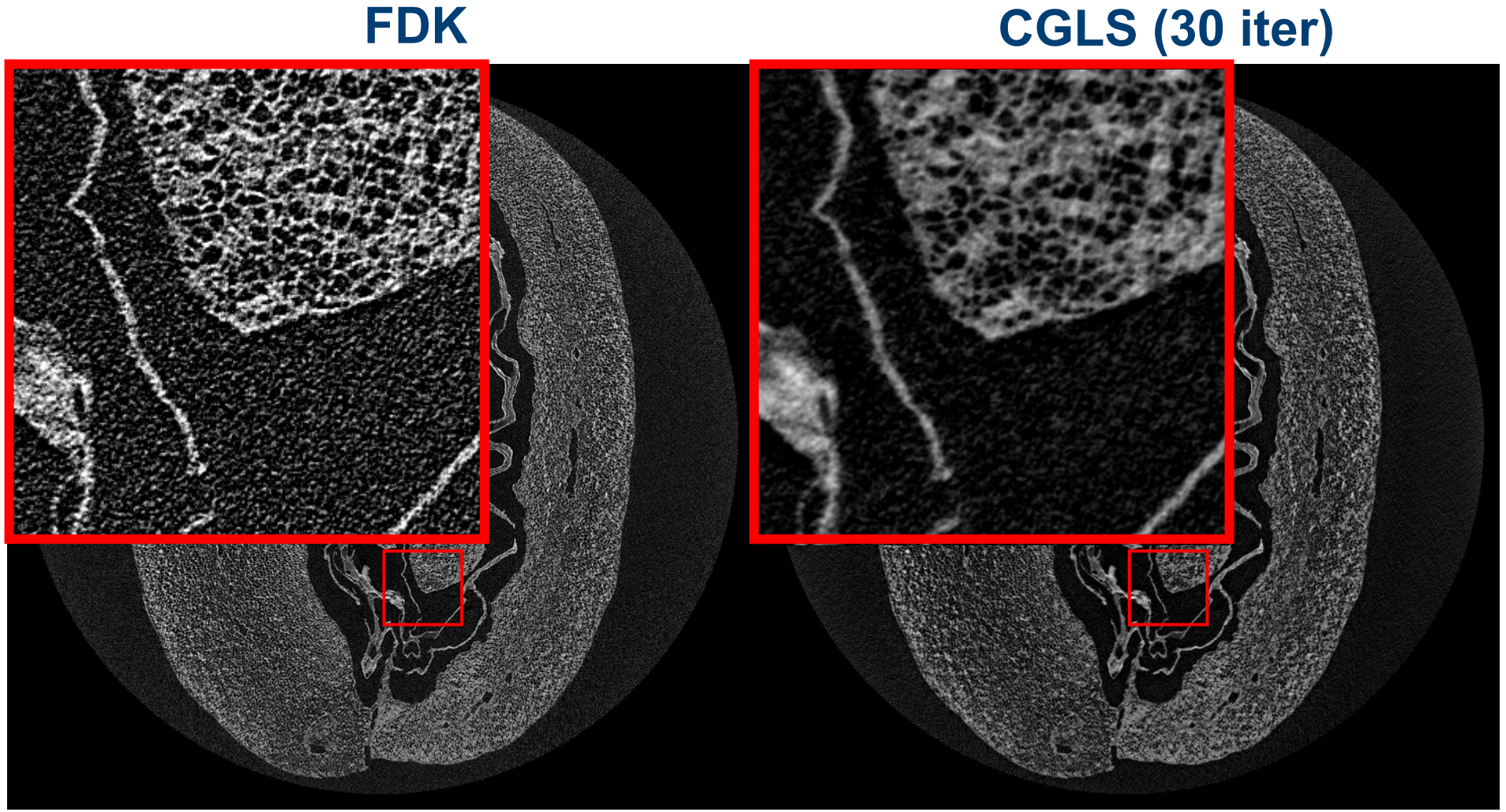
Prior image regularized methods
- PICCS

Proximal Methods
- FISTA
- Accelerated FISTA



Wilbers, A., **Biguri, A.**, Schillings, J., & Loos, J. (2019).
Application of iterative reconstruction algorithms to mitigate CT-artefacts when measuring fiber reinforced polymer materials.
Polymer, *177*, 120-130

MicroCT: Roasted Coffee Beam



3340x3340x900 coffee bean, 3.6 micrometres per pixel
4000x2000x2100 X-ray projections

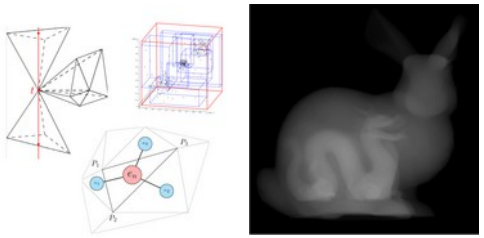
45:00h scan time
04:21h reconstruction time

Arbitrarily large tomography with iterative algorithms on multiple GPUs using the TIGRE toolbox

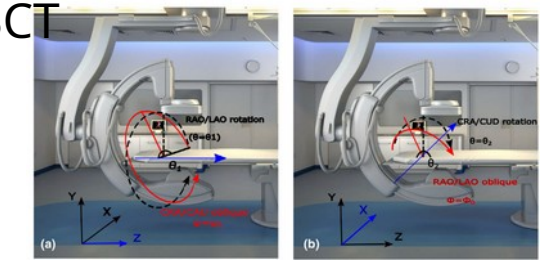
Ander Biguri [a], Reuben Lindroos [a], Robert Bryll [b], Hossein Towsyfyan [a], Hans Deyhle [a], Ibrahim El khalil Harrane [a], Richard Boardman [c], Mark Mavrogordato [c], Manjit Dosanjh [d], Steven Hancock [d], Thomas Blumensath [a]

Other related work

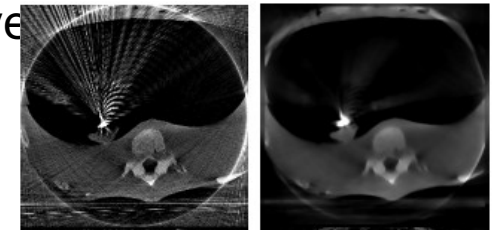
Tomography on tetrahedra meshes on GPUs



Scanning trajectory optimization in C-arm CBCT

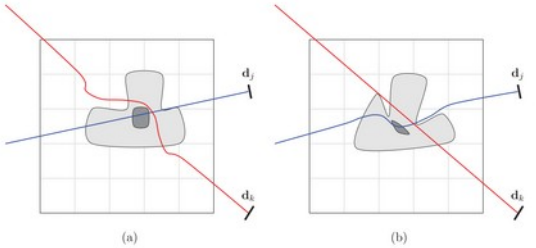


Prior Image Constrained regularization in Krylov solve

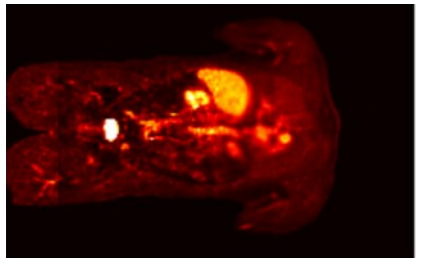


$$\boldsymbol{x}_k = \min_{\boldsymbol{x}} \left\| \begin{bmatrix} \mathbf{A} \\ \alpha \mathbf{W}_k^{(1)} D \\ \lambda \mathbf{W}_k^{(2)} D \end{bmatrix} \boldsymbol{x} - \begin{bmatrix} \boldsymbol{b} \\ \mathbf{0} \\ \lambda \mathbf{W}_k^{(2)} D \boldsymbol{x}_p \end{bmatrix} \right\|_2^2 = \min_{\boldsymbol{x}} \left\| \tilde{\mathbf{A}} \boldsymbol{x} - \tilde{\boldsymbol{b}} \right\|_2^2,$$

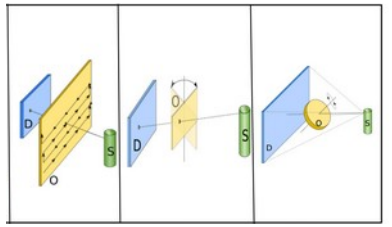
Motion Compensated CBCT for radiation therapy



Dynamic-whole Body Positron Emission Tomography



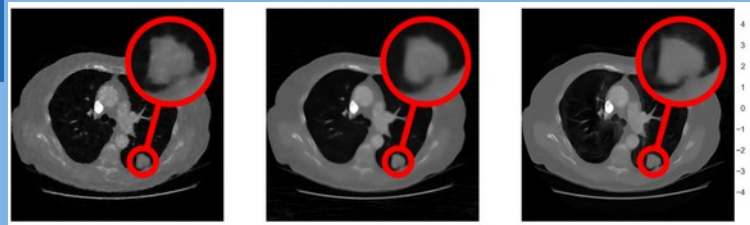
Laminographic scanning procedures in the lab



and others....

Data-driven inverse problems

From maths to applications



Data-Driven inverse problems

- One can learn parts (or all) of the variational formulation

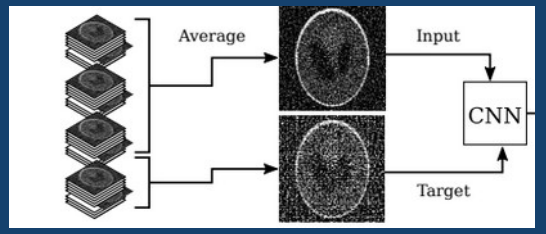
$$\hat{\mathbf{x}} = \underset{\mathbf{x}}{\operatorname{argmin}}\ L$$

- The field has exploded in applied maths and computational imaging in the last 8 years

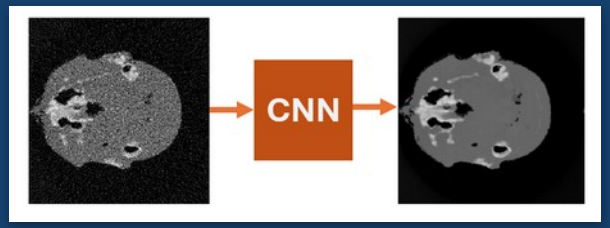
- Theoretical performance of these methods is much superior to “classical” approaches

Data Driven Tomographic Reconstruction

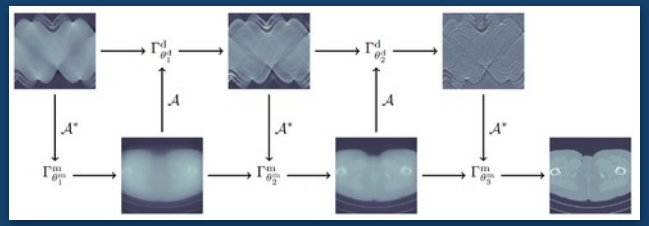
Self/Unsupervised



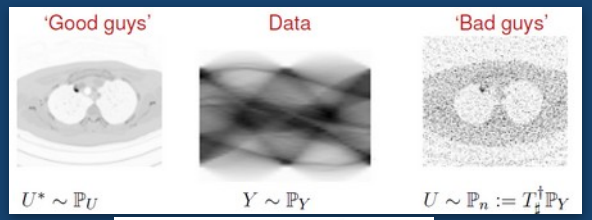
Post-Processing



Unrolled Iterative Methods



Learning a regulariser



$$x_{reg} = \arg\min_x \|Ax - b\|_2^2 + \lambda R(x)$$

Plug and Play

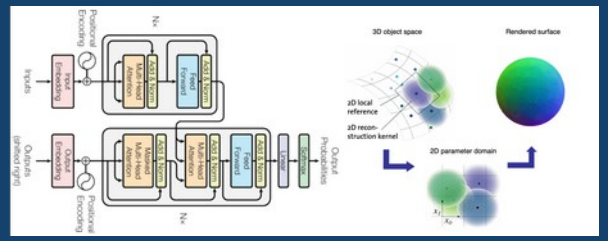
$$\mathbf{x}^{k+1} = \operatorname*{argmin}_{\mathbf{x}} D(\mathbf{Ax}, \mathbf{b}) + \lambda \|\mathbf{x} - \mathbf{v}^k + \mathbf{d}^k\|$$

Learned Denoiser

$$\mathbf{v}^{k+1} = \operatorname*{argmin}_{\mathbf{v}} R(\mathbf{v}) + \lambda \|\mathbf{v} - \mathbf{x}^{k+1} + \mathbf{d}^k\|$$

$$\mathbf{d}^{k+1} = \mathbf{d}^k - (\mathbf{x}^{k+1} - \mathbf{v}^{k+1})$$

Other ML methods



Current challenges for translation

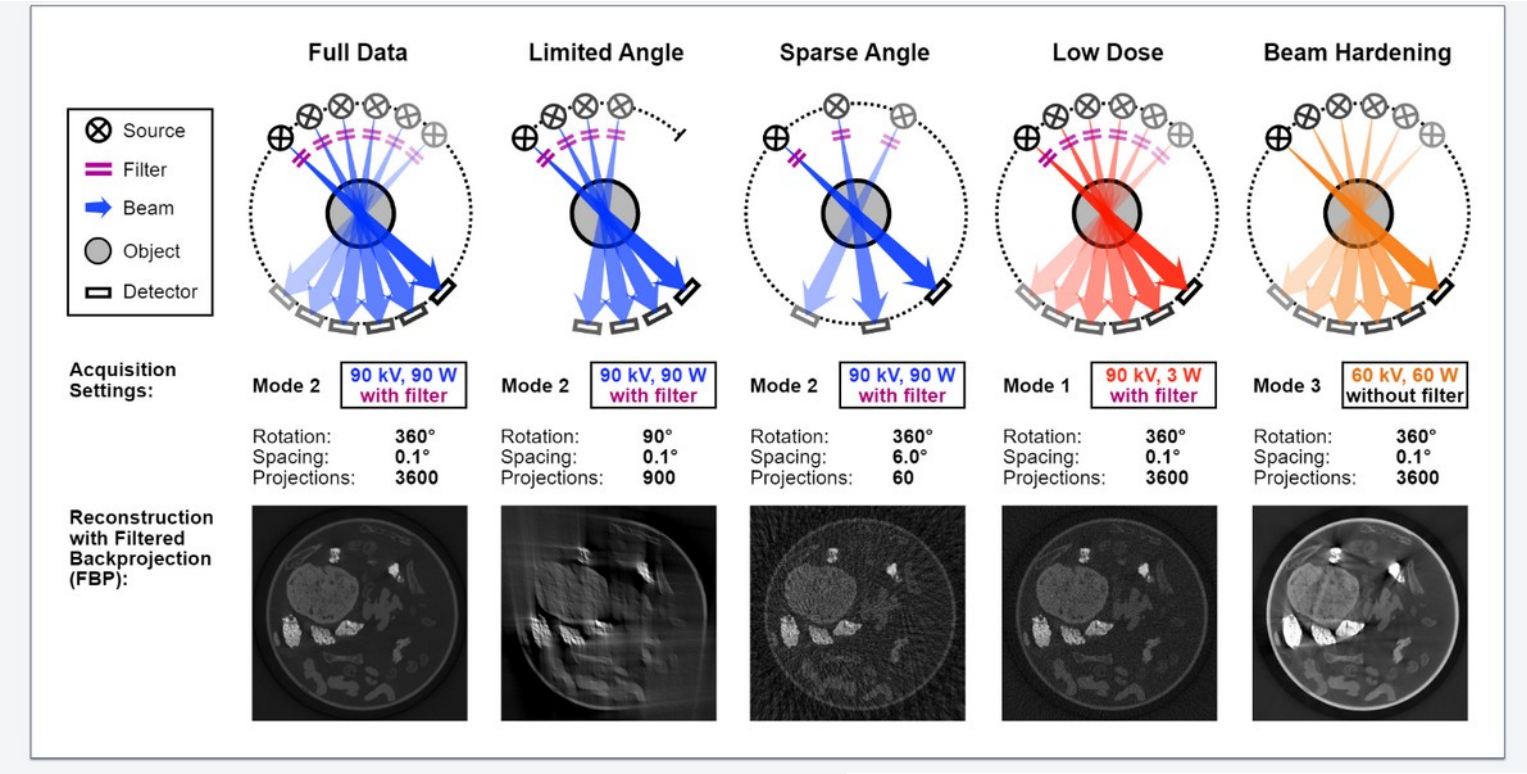
- Data-driven recon research is very slowly moving to real data use. Many challenges still
- Current challenges:
  - Lack of comparability/reproducibility
  - Lack of understanding/uncertainty quantification
  - Vague evaluation criteria → What is a “good” result?
  - (Almost) never been tested with real data
  - (Almost) never used in 3D data
  - Field is new → best methods still to come

- What am I* doing about his?

*Researchers working with

Lack of comparability

Compare 12 ML-methods in 9 different CT cases.



Benchmarking learned algorithms for computed tomography image reconstruction tasks

Maximilian B. Kiss[1, #], Ander Biguri[2, #], Zakhar Shumaylov[2], Ferdia Sherry[2], K. Joost Batenburg[3], Carola-Bibiane Schönlieb[2] and Felix Lucka[1]

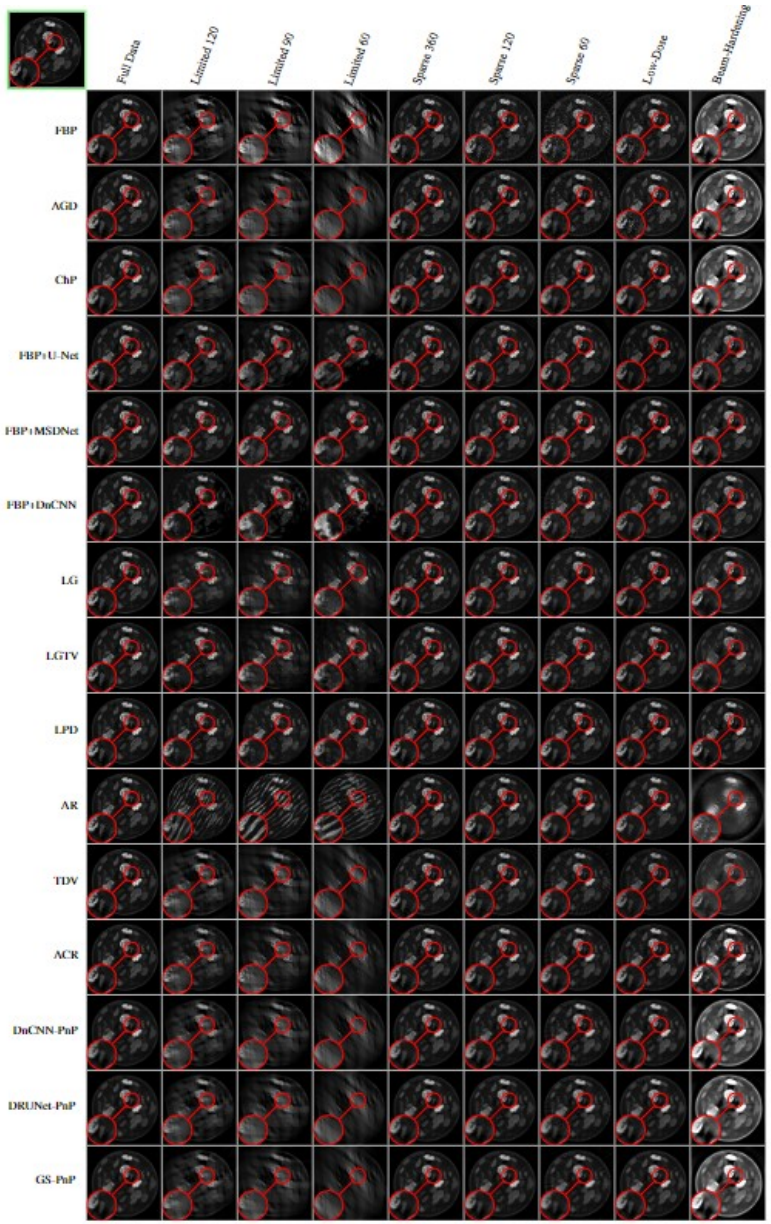
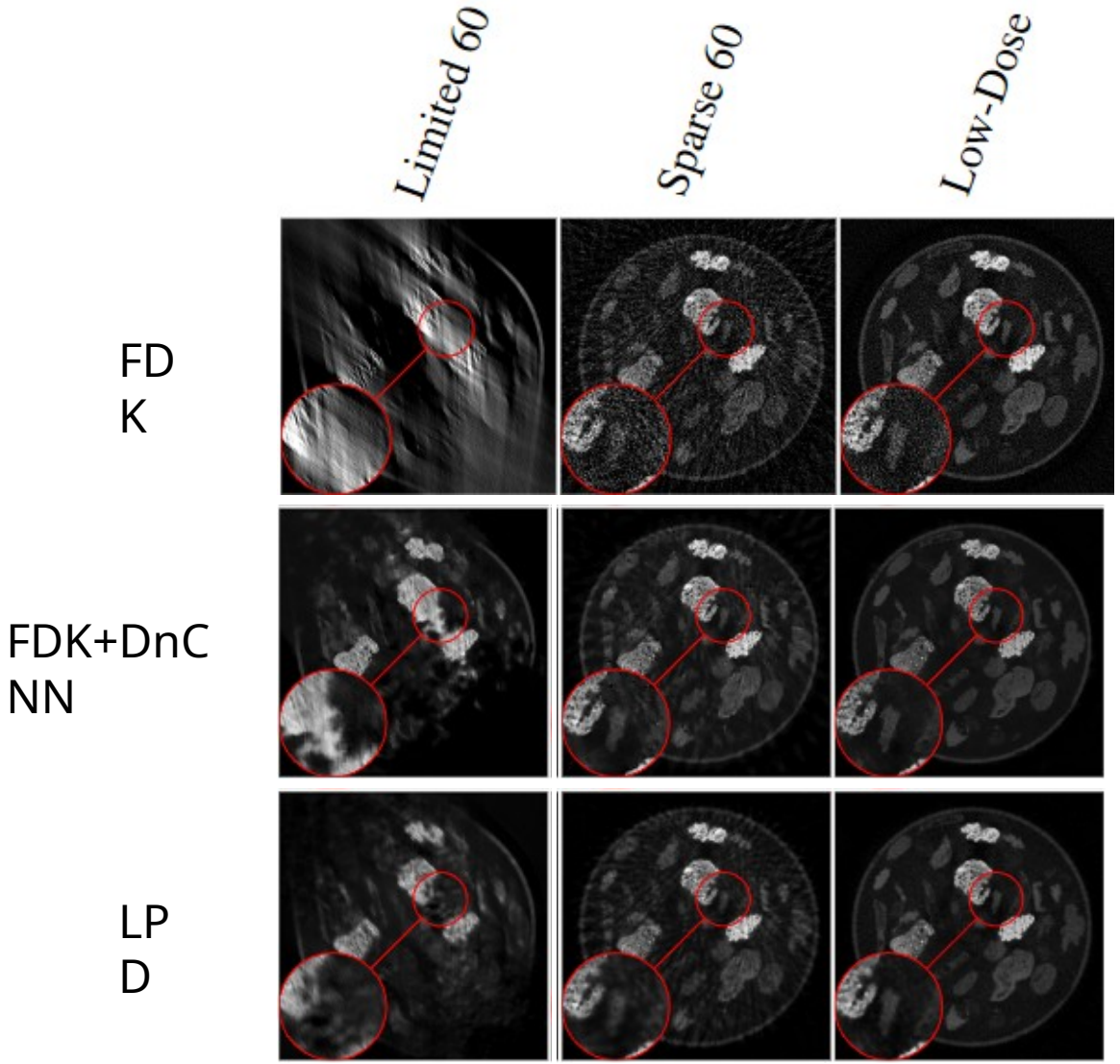


Table 3: Quantitative analysis of all evaluated methods with respect to PSNR and SSIM on their performance in the CT image reconstruction tasks.

Method	Metric	CT Image Reconstruction Task								
		Full Data	Limited 120	Limited 90	Limited 60	Sparse 360	Sparse 120	Sparse 60	Low-Dose	Beam-Hardening
Classical Methods										
FBP	SSIM	0.7463 ± 0.0296	0.3418 ± 0.0354	0.2369 ± 0.0323	0.1557 ± 0.0288	0.2947 ± 0.0453	0.1231 ± 0.0225	0.0611 ± 0.0112	0.0838 ± 0.0212	0.3367 ± 0.0464
	PSNR	35.0285 ± 2.0907	22.4188 ± 1.9240	19.4251 ± 1.9405	16.9057 ± 1.9579	24.9674 ± 2.0415	19.8769 ± 2.0124	16.6451 ± 1.9972	18.6437 ± 2.0508	14.5594 ± 1.9056
AGD	SSIM	0.7753 ± 0.0380	0.4904 ± 0.0550	0.4411 ± 0.0555	0.4146 ± 0.0559	0.3867 ± 0.0563	0.4142 ± 0.0630	0.4333 ± 0.0664	0.0727 ± 0.0182	0.3483 ± 0.0650
	PSNR	35.1006 ± 2.2801	25.7508 ± 1.9690	24.1128 ± 1.9528	22.8848 ± 1.9531	26.9629 ± 2.0444	27.5127 ± 1.9948	27.2796 ± 1.9553	17.7062 ± 2.0517	14.4294 ± 1.9255
ChP	SSIM	0.7689 ± 0.0498	0.5923 ± 0.0550	0.5194 ± 0.0547	0.4658 ± 0.0510	0.6998 ± 0.0685	0.6712 ± 0.0718	0.5952 ± 0.0728	0.6820 ± 0.0749	0.4492 ± 0.0616
	PSNR	34.3251 ± 1.9703	26.3646 ± 1.9443	24.4392 ± 1.9321	22.7556 ± 1.9428	32.9158 ± 1.9739	31.7980 ± 1.9798	29.8020 ± 1.9326	31.9637 ± 1.9824	15.0090 ± 1.9094
Post-Processing Methods										
FBP+U-Net	SSIM	0.6499 ± 0.0681	0.7251 ± 0.0519	0.6338 ± 0.0659	0.5892 ± 0.0678	0.7449 ± 0.0801	0.7518 ± 0.0657	0.7728 ± 0.0592	0.7632 ± 0.0780	0.6336 ± 0.0760
	PSNR	32.6998 ± 1.9512	28.7931 ± 2.0052	27.3875 ± 2.0441	23.8511 ± 2.8370	30.4766 ± 3.4844	26.0785 ± 6.3779	19.5421 ± 9.9613	27.6772 ± 3.0133	28.0629 ± 3.8439
FBP+MSDNet	SSIM	0.8481 ± 0.0384	0.7840 ± 0.0641	0.7695 ± 0.0579	0.7148 ± 0.0726	0.8392 ± 0.0473	0.7993 ± 0.0650	0.7626 ± 0.0789	0.7253 ± 0.0864	0.7991 ± 0.0665
	PSNR	33.2999 ± 1.9492	30.7850 ± 1.9784	29.0111 ± 1.9740	27.4624 ± 1.9661	33.1188 ± 1.9820	32.2993 ± 1.9928	30.9931 ± 1.9875	31.6910 ± 1.9559	31.4389 ± 2.0988
FBP+DnCNN	SSIM	0.8324 ± 0.0403	0.5829 ± 0.0521	0.5661 ± 0.0528	0.5174 ± 0.0559	0.7864 ± 0.0618	0.6701 ± 0.0864	0.6180 ± 0.0796	0.7127 ± 0.0806	0.6328 ± 0.0751
	PSNR	32.2575 ± 1.9506	20.4433 ± 2.5345	23.4066 ± 2.2764	21.2132 ± 2.5140	31.7575 ± 2.0213	29.1817 ± 2.3053	28.6079 ± 2.3389	29.7645 ± 1.9748	28.2405 ± 2.0374
Learned / Unrolled Iterative Methods										
LG	SSIM	0.7498 ± 0.0708	0.6740 ± 0.0652	0.5746 ± 0.0655	0.5378 ± 0.0706	0.7846 ± 0.0628	0.6795 ± 0.0790	0.6428 ± 0.0777	0.6685 ± 0.0772	0.6025 ± 0.0747
	PSNR	32.4409 ± 1.9527	28.1639 ± 1.9380	26.3188 ± 1.9508	24.7228 ± 1.9630	32.5946 ± 1.9764	31.1950 ± 1.9658	29.9360 ± 1.9603	30.4679 ± 1.9945	28.4148 ± 1.9584
LGTV	SSIM	0.8221 ± 0.0532	0.6804 ± 0.0647	0.5590 ± 0.0639	0.5091 ± 0.0643	0.7811 ± 0.0659	0.7100 ± 0.0745	0.7081 ± 0.0689	0.7008 ± 0.0797	0.6656 ± 0.0774
	PSNR	33.3312 ± 1.9493	28.4131 ± 1.9300	26.0867 ± 1.9545	25.0225 ± 1.9687	32.9404 ± 1.9666	31.4072 ± 1.9647	29.9021 ± 1.9357	30.9991 ± 1.9328	29.5372 ± 2.0008
LPD	SSIM	0.8447 ± 0.0370	0.8296 ± 0.0410	0.8049 ± 0.0444	0.7724 ± 0.0571	0.8433 ± 0.0479	0.8300 ± 0.0500	0.8206 ± 0.0508	0.8282 ± 0.0519	0.8352 ± 0.0568
	PSNR	33.3086 ± 1.9466	31.1723 ± 1.9607	29.2534 ± 1.9941	28.0734 ± 1.9589	33.3809 ± 1.9513	32.7032 ± 1.9685	32.0583 ± 1.9789	32.6685 ± 1.9656	33.1382 ± 1.9759
Learned Regularizer Methods										
AR	SSIM	0.8196 ± 0.0385	0.6869 ± 0.0505	0.6100 ± 0.0543	0.5742 ± 0.0620	0.8309 ± 0.0447	0.8117 ± 0.0553	0.7949 ± 0.0595	0.8039 ± 0.0505	0.3125 ± 0.0479
	PSNR	32.3583 ± 1.9621	23.8496 ± 2.1578	21.1830 ± 2.1772	22.2350 ± 2.0260	32.8067 ± 1.9714	32.1030 ± 1.9577	30.8378 ± 1.9532	31.0472 ± 1.9763	19.5692 ± 1.9612
TDV	SSIM	0.7282 ± 0.0652	0.5940 ± 0.0595	0.5459 ± 0.0572	0.5282 ± 0.0584	0.6815 ± 0.0736	0.6235 ± 0.0741	0.5725 ± 0.0728	0.6047 ± 0.0815	0.5494 ± 0.0635
	PSNR	33.1204 ± 1.9496	26.3233 ± 1.9315	24.8127 ± 1.9399	23.3939 ± 1.9662	32.2673 ± 1.9641	30.6585 ± 1.9357	28.9451 ± 1.8995	28.2429 ± 1.9433	25.7832 ± 2.0109
ACR	SSIM	0.8518 ± 0.0362	0.7114 ± 0.0543	0.6575 ± 0.0541	0.5515 ± 0.0529	0.8271 ± 0.0494	0.8074 ± 0.0539	0.7849 ± 0.0524	0.8163 ± 0.0522	0.6742 ± 0.0505
	PSNR	33.7131 ± 1.9450	27.1792 ± 1.9441	25.3342 ± 1.9442	23.4915 ± 1.9539	33.1537 ± 1.9632	31.9181 ± 1.9666	30.5147±1.9338	32.2621 ± 1.9689	18.5708 ± 1.7824
Plug-and-Play Methods										
DnCNN-PnP	SSIM	0.8585 ± 0.0402	0.7617 ± 0.0410	0.6981 ± 0.0441	0.6200 ± 0.0475	0.8405 ± 0.0432	0.8021 ± 0.0465	0.7637 ± 0.0484	0.7795 ± 0.0523	0.5989 ± 0.0375
	PSNR	32.8506 ± 1.9306	26.9997 ± 1.9330	25.0658 ± 1.9248	23.4108 ± 1.9521	32.4627 ± 1.9309	31.3847 ± 1.9271	29.9350 ± 1.8980	31.3986 ± 1.9424	15.4667 ± 1.9107
DRUNet-PnP	SSIM	0.8573 ± 0.0405	0.7634 ± 0.0411	0.7002 ± 0.0443	0.6149 ± 0.0511	0.8398 ± 0.0433	0.8000 ± 0.0465	0.7658 ± 0.0498	0.7984 ± 0.0517	0.5945 ± 0.0375
	PSNR	32.8935 ± 1.9327	27.0262 ± 1.9334	25.0829 ± 1.9254	23.4362 ± 1.9520	32.5065 ± 1.9324	31.3949 ± 1.9266	29.9518 ± 1.9085	31.5762 ± 1.9480	15.4543 ± 1.9102
GS-PnP	SSIM	0.7856 ± 0.0590	0.6396 ± 0.0670	0.5668 ± 0.0663	0.4989 ± 0.0625	0.7622 ± 0.0628	0.7588 ± 0.0684	0.6937 ± 0.0701	0.7727 ± 0.0683	0.5131 ± 0.0562
	PSNR	32.5734 ± 1.9331	26.3318 ± 1.9328	24.5129 ± 1.9282	22.8225 ± 1.9481	32.1977 ± 1.9353	31.2728 ± 1.9370	29.5791 ± 1.8960	31.7931 ± 1.9444	15.3466 ± 1.9128


Limited 60
Sparse 60
Low-Dose
FDK
FDK+DnCNN
LPD

(Almost) Never used with real data

- Most math literature simplifies noise models in CT
  - OK for math paper
  - Not OK for applications

- We can interpret many of the ML models as learning the distribution of the noise -> If we simulate this distribution, it better be good!

Is training with simulated noise enough?

Simulate complex (yet computable) CT noise on “clean” data to the same SNR as measured noisy data.

- Electronic noise
- Photon noise
- Flat fields
- Detector Cross talk
- Digital quantitation errors

Train denoiser Unet on noisy data. Both simulated and real.

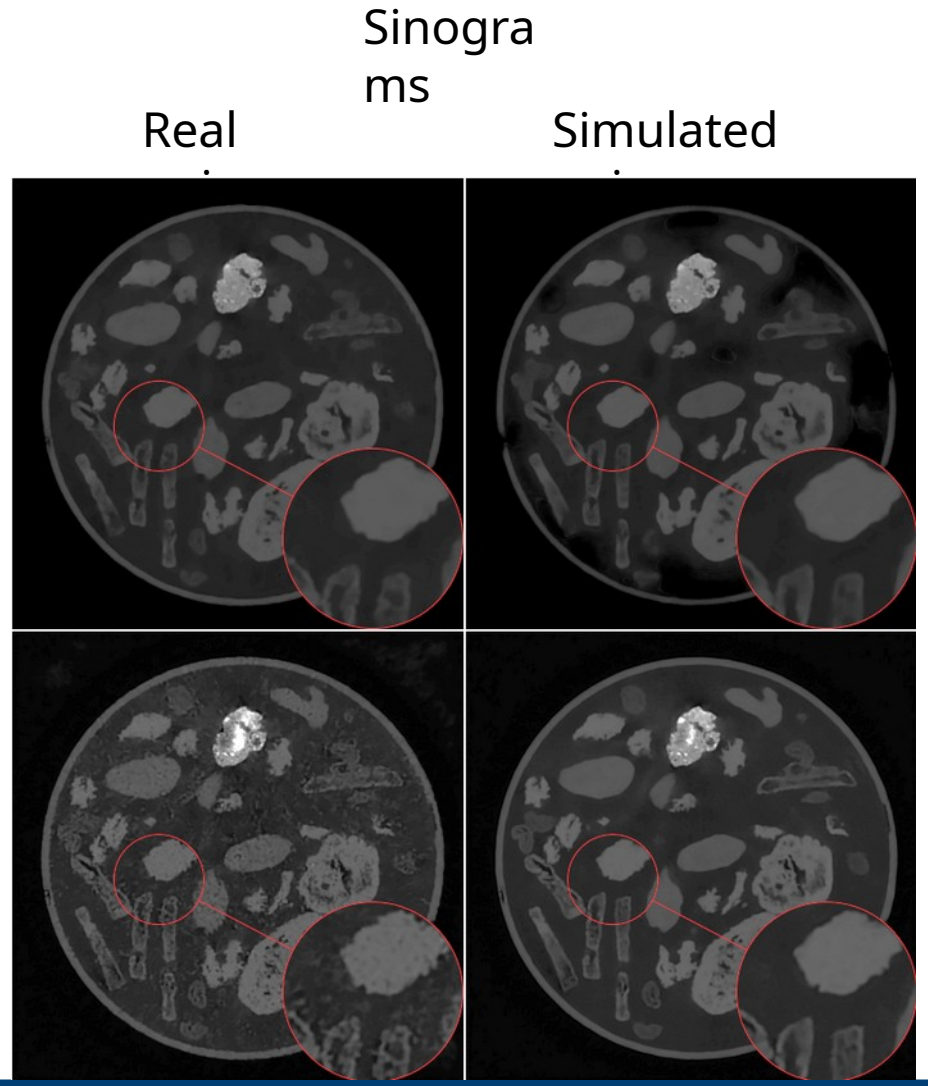
Test both Unets in both datasets.

Do we need to train in real noise?

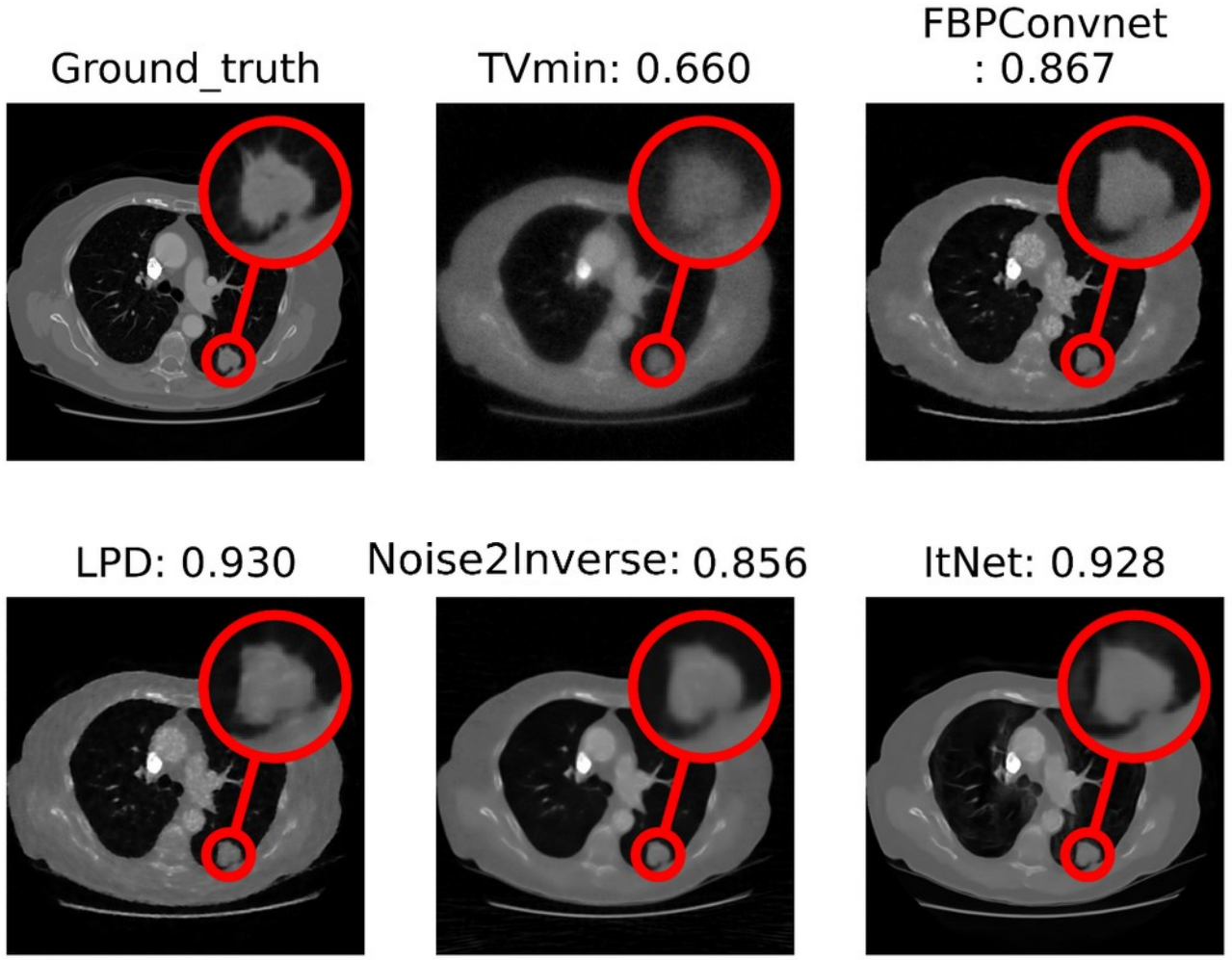
FBP+U-Net (2)	Experimental Noisy Data	SSIM PSNR	**0.8161 ± 0.0592** 29.8398 ± 2.1834	0.7466 ± 0.0681 28.0435 ± 2.0848
FBP+U-Net (1)	Simulated Noisy Data	SSIM PSNR	0.5957 ± 0.0844 26.8841 ± 3.4800	0.6693 ± 0.0820 28.8134 ± 3.9418

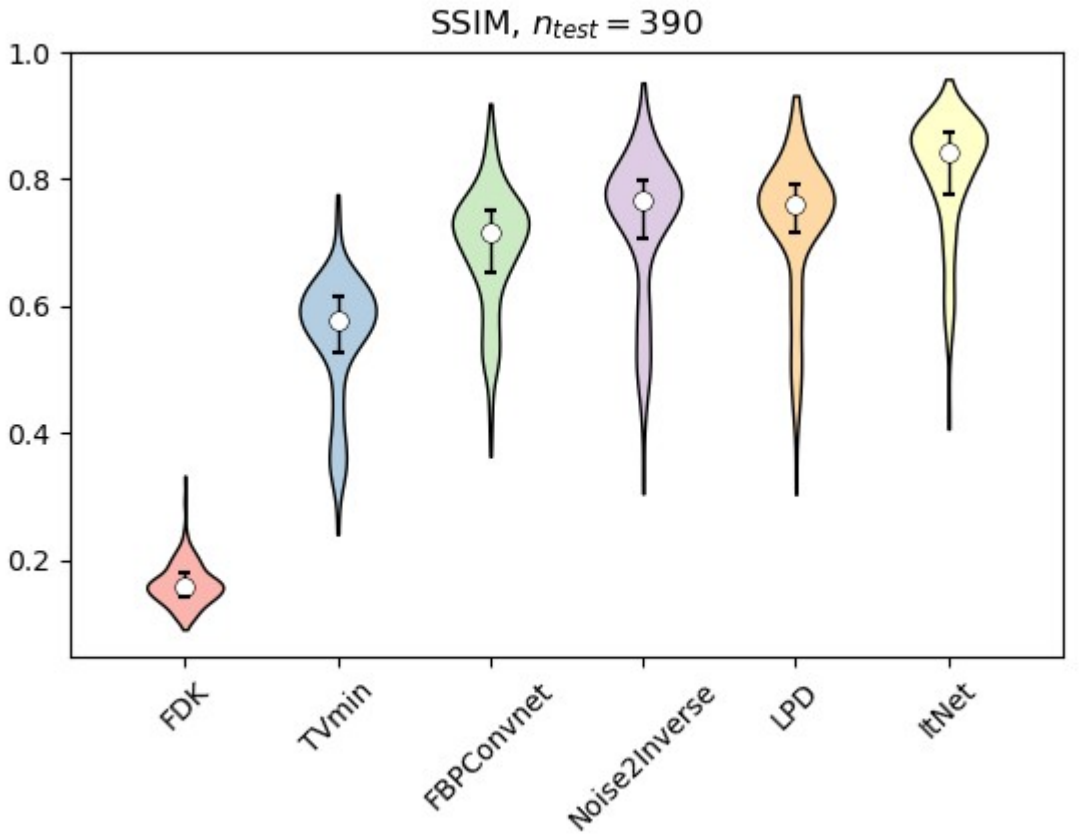
Trained in real noise

Trained in simulated noise

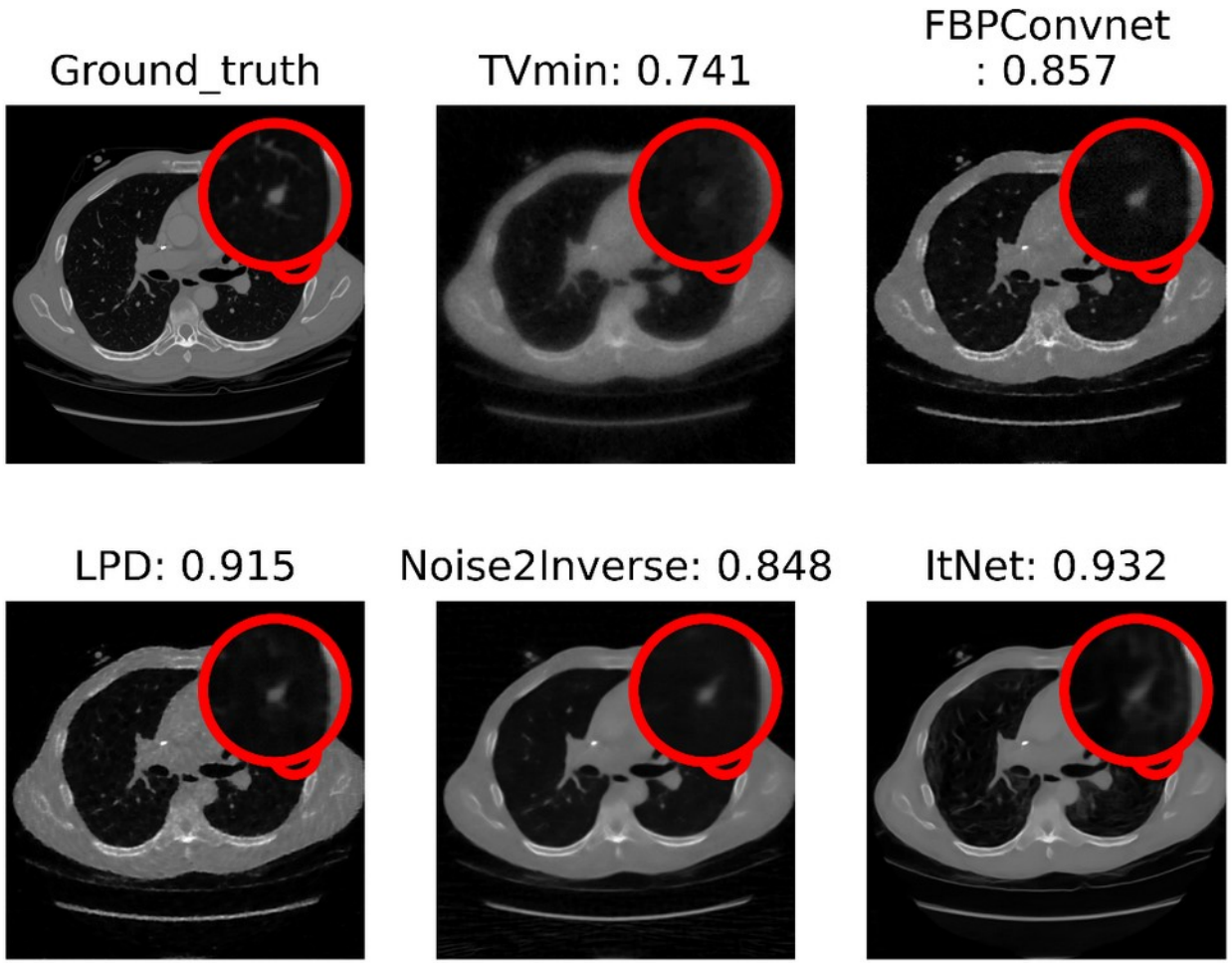


Vague evaluation criteria → What is a “good” result?





But… which one is better?



A Study of Why We Need to Reassess Full Reference Image Quality Assessment with Medical Images

Original Paper | Open access | Published: 24 March 2025

(2025) Cite this article

Download PDF

You have full access to this open access article

Anna Breger, Ander Biguri, Malena Sabaté Landman, Ian Selby, Nicole Amberg, Elisabeth Brunner, Janek Gröhl, Sepideh Hatamikia, Clemens Karner, Lipeng Ning, Sören Dittmer, Michael Roberts, AIX-COVNET Collaboration & Carola-Bibiane Schönlieb



Andrea **Sainz**

Can we train learned methos to produce images that are better for our desired task?

But… which one is better?

Geometry-Aware Diffusion Posterior Sampling for Sparse-View and Limited-Angle CT

Honglei Brinkmann, Carola-Bibiane Schönlieb, Ander Biguri
Department of Applied Mathematics and Theoretical Physics, University of Cambridge, Cambridge, UK.



Honglei **Brinkmann**

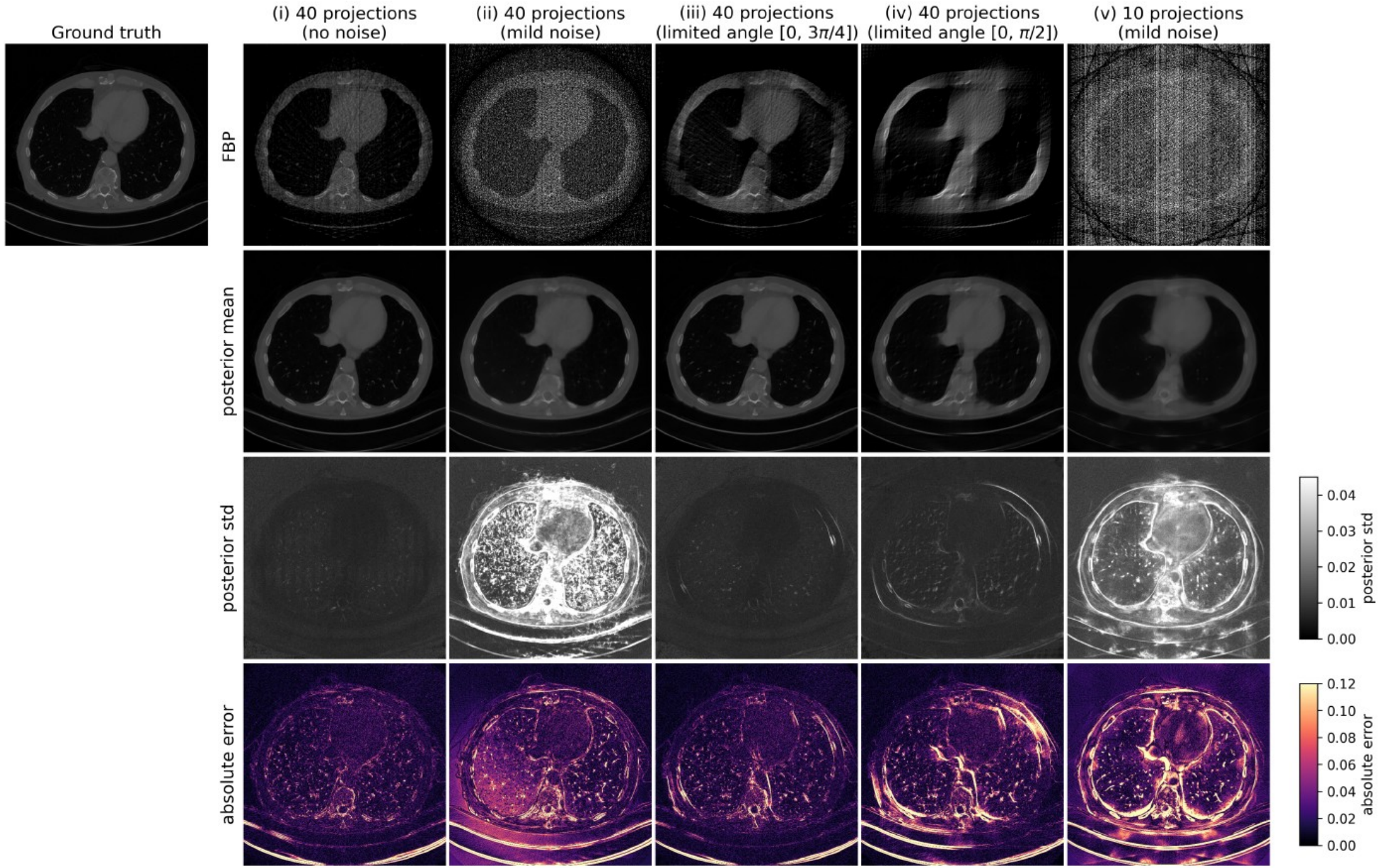
- In Arxiv this week (hopefully)

- Also tackles uncertainty!



Lack of understanding/uncertainty quantification

- Interpretability (in its broader meaning) is essential for medical applications.
  - What does the model do?
  - When does it fail?
  - Confidence in the algorithm?
  - Confidence in the results? (uncertainty quantification)

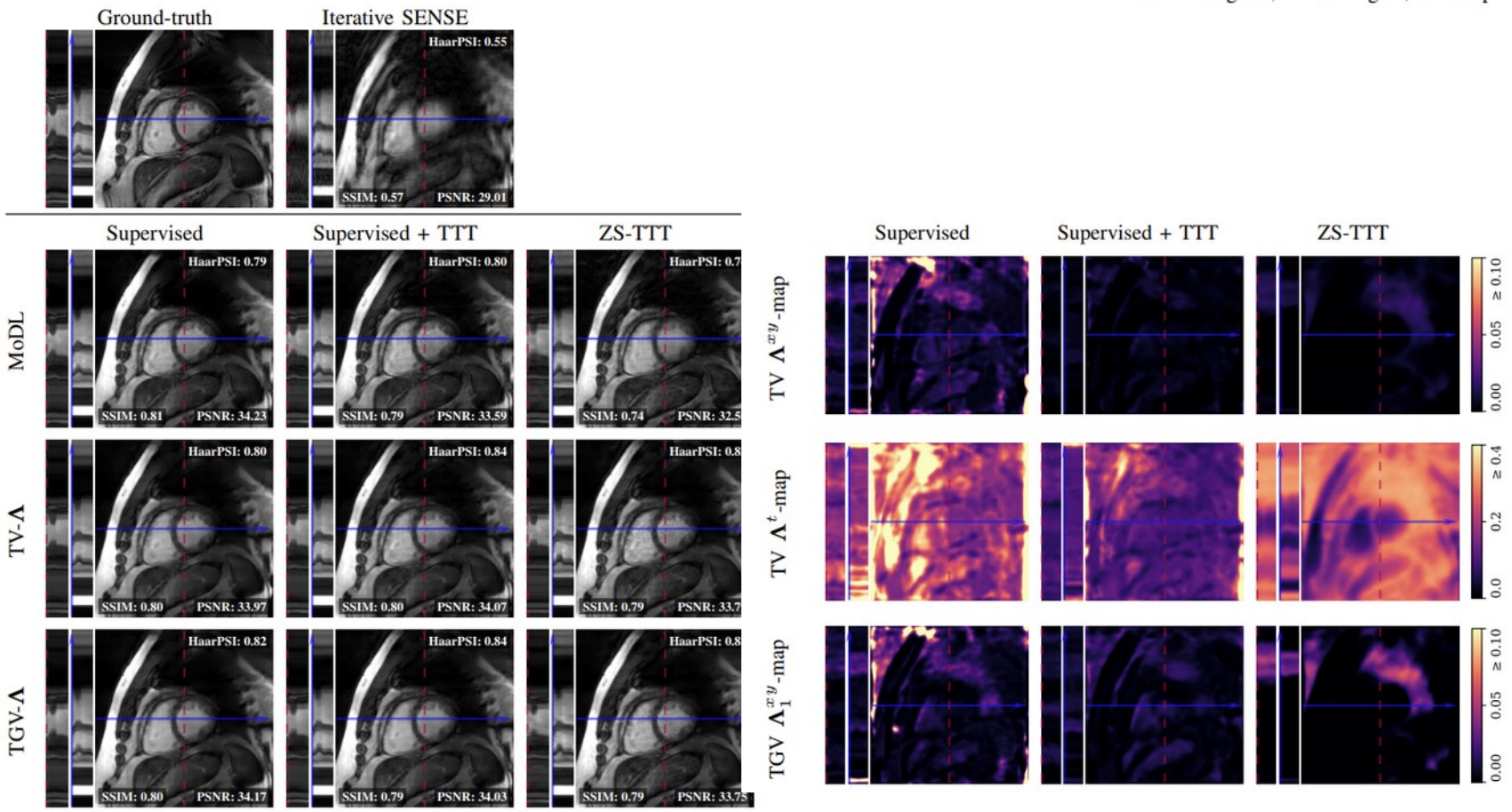

Ground truth
(i) 40 projections (no noise)
(ii) 40 projections (mild noise)
(iii) 40 projections (limited angle [0, 3π/4])
(iv) 40 projections (limited angle [0, π/2])
(v) 10 projections (mild noise)
FBP
posterior mean
posterior std
absolute error
0.04
0.03
0.02
0.01
0.00
posterior std
0.12
0.10
0.08
0.06
0.04
0.02
0.00
absolute error

We can change interpretability by only learning interpretable things

- E.g. if we learn to predict regularization hyperparameters, we understand what the results mean

DREAM: Deep-Reparametrization of Adaptive Regularization Maps for Fast Zero-Shot Self-Supervised Learning

Thanh Trung Vu, Ander Biguri, Christoph Kolbitsch, Luca Calatroni, Kostas Papafitsoros, Andreas Kofler



Performs almost as good as state of the art big ML models, but its unsupervised and interpretable

(Almost) never used in 3D data

- I ran Grand Challenge in 2024 ICASSP



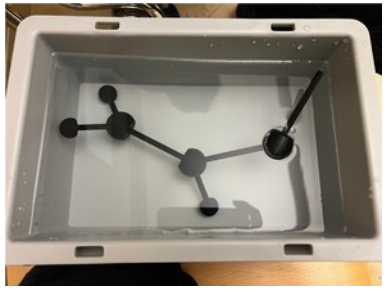
Advancing the Frontiers of Deep Learning for Low-Dose 3D Cone-Beam CT Reconstruction

Publisher: IEEE | Cite This | PDF

Ander Biguri ; Subhadip Mukherjee ; Xuzhi Zhao ; Xi Liu ; Xinyi Wang ; Rui Yang **All Authors**

- Working on acquiring the first 3D CBCT ML-sized dataset

  Sepideh
  Hatamikia
  S M Ragib **Islam**

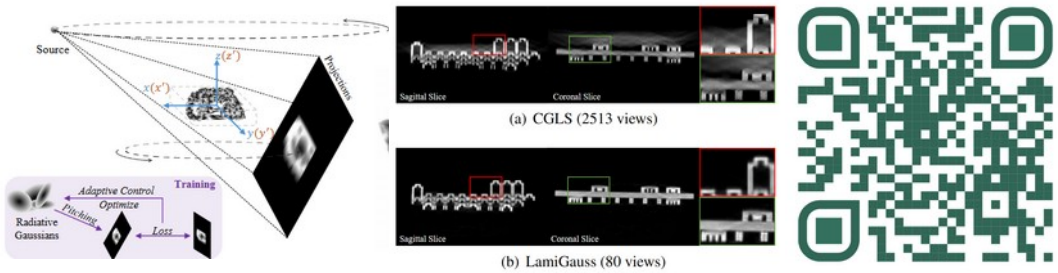


- Exploring ML methods that tolerate 3D CBCT

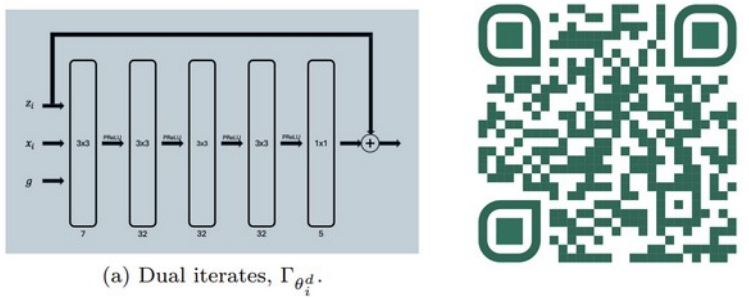
  Prachi
  Sharma
  S M Ragib
  Islam

Field is new → best methods still to come

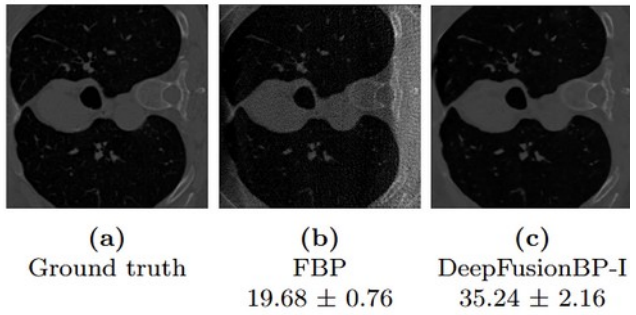
Gaussian Splatting - unsupervised



Neural ODEs + unrolling



FBP + CNNs



Andrea
Sainz

etc

LION V 0.1

Learned Iterative Optimization Networks

Tomographic Reconstruction
Machine Learning
Reproducibility

=> Make CT tools accessible to ML researchers
Make ML tools accessible to CT researchers <=

=> Research unification and sharing

=> Long term reproducibility

LION V 0.1
Learned Iterative Optimization Networks

Tomographic Reconstruction
Machine Learning
Reproducibility

github.com/CambridgeCIA/LION

Implemented models

DataSets

Problems

Experiments

clinicalCTRecon()
LowDoseCTRecon()
ExtremeLowDoseCTRecon()
LimitedAngleCTRecon()
SparseAngleCTRecon()
…

Models

Post-processing
- FBPConvNet
- DnCNN
- MSD-Net
- Unet
- DnCNN
- DRUnet

Iterative Unrolled
- LPD
- LGD
- LGD-TV

Learned Regularizer:
- AR
- ACR
- TDV

Plug and Play
- Any post-processing
- Gradient-Step Denoiser

Algorithm-based
- DeepFBP
- FusionFBP

Solvers

SupervisedSolver()
Noise2InverseSolver()
GaussianDenoiserSolver()
EquivariantSolver()
AdversarialRegularizerSolver()

Reconstructor

Post_processing

PnP
- Forward/ backward splitting

Eval

PSNR
SSIM
HaarPSI

Summary

$$\hat{\mathbf{x}} = \underset{\mathbf{x}}{\operatorname{argmin}}\ D(\mathbf{Ax},$$

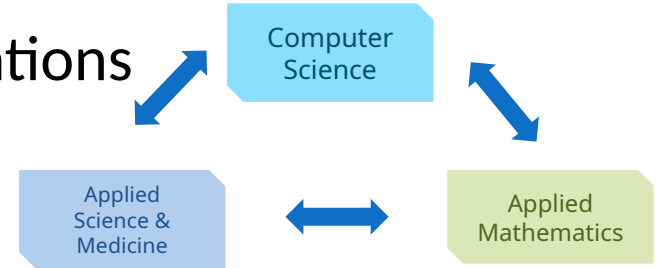
- AI methods in inverse problems are super powerful, but they are still too many unknowns to make them trivial to use in experimental cases. A lot of real work to do outside frontiers of AI.

- Good Software tools lead to research innovations



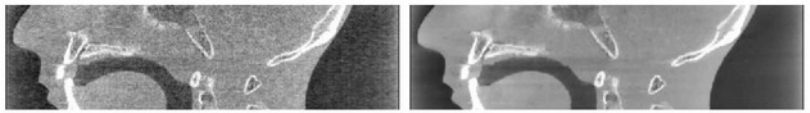
- There is much to share between applications of tomography, and they can inform much of the mathematical development directions

- Still quite a lot to do to make advances work in real life applications



Questions

CAT: Computational Applied T...



The CAT is a research group on translational methods for tomography lead by Ander Biguri.

The research focus is threefold:

- Developing **mathematical optimization algorithms** based on both machine learning based and traditional optimization for inverse problems in tomography (applied maths).
- Developing **computational methods**, high performance computing solutions, machine learning models and software packages for the algorithms above (computer science).
- **Applying these methods into real datasets** and applications of tomography, and proposing novel uses of tomography enabled by the previous two points (applied science & medicine).

Find a full list of publications in the Publications page and the Gallery of research images for more information.

The CAT currently also maintains two open source packages for reconstruction, particularly for all modalities of CT:

- TIGRE toolbox 🐯: A toolbox for easy to use and fast tomographic reconstruction of real datasets
- LION toolbox 🦁 (in development): A toolbox for reliable and reproducible Machine Learning based tomographic reconstruction.



ander biguri .github .io



Andrea Sainz Bear



Honlgei Brinkmann



Vu Thanh Trung (Troy)



Dr Ander Biguri



Alvaro Esposito Martinez

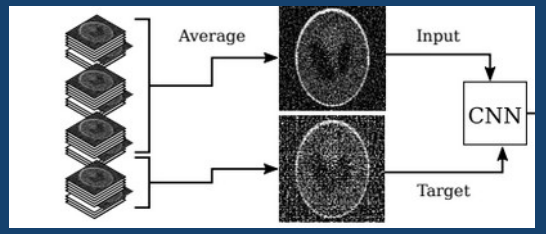


Georg Francis Barlaup Lovric

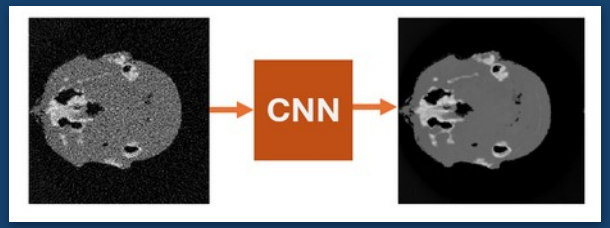


Data Driven Tomographic Reconstruction

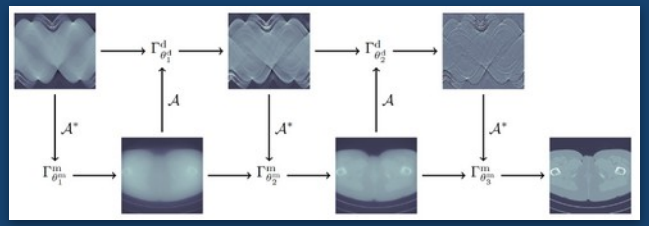
Self/Unsupervised



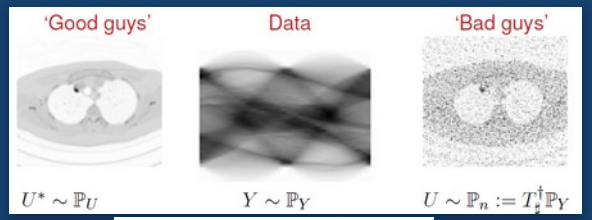
Post-Processing



Unrolled Iterative Methods



Learning a regulariser



$$x_{reg} = \arg\min_x \|Ax - b\|_2^2 + \lambda R(x)$$

Plug and Play

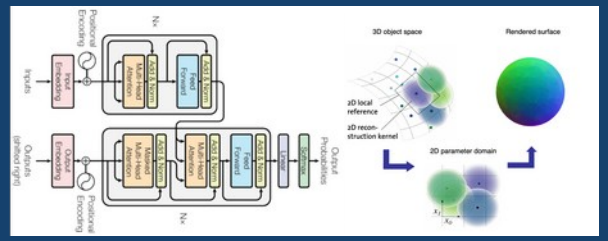
$$\mathbf{x}^{k+1} = \operatorname*{argmin}_{\mathbf{x}} D(\mathbf{Ax}, \mathbf{b}) + \lambda \|\mathbf{x} - \mathbf{v}^k + \mathbf{d}^k\|$$

Learned Denoiser

$$\mathbf{v}^{k+1} = \operatorname*{argmin}_{\mathbf{v}} R(\mathbf{v}) + \lambda \|\mathbf{v} - \mathbf{x}^{k+1} + \mathbf{d}^k\|$$

$$\mathbf{d}^{k+1} = \mathbf{d}^k - (\mathbf{x}^{k+1} - \mathbf{v}^{k+1})$$

Other ML methods



Example: Gradient descend

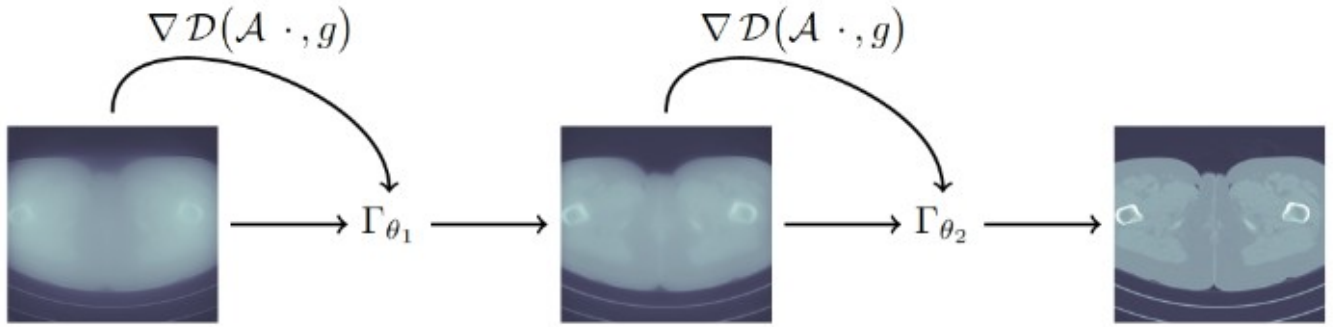
$$\mu_0 = 0$$

$$\mu_{k+1} = \mu_k + \gamma \nabla \|A\mu_k - p\|^2 \qquad k = 0, 1, \ldots, N$$

Learned Gradient descend

$$\mu_0 = 0$$

$$\mu_{k+1} = \Gamma_{\theta_k}(\mu_k, \nabla\|A\mu_k - p\|^2) \quad k = 0, 1, \ldots, N$$



Adler, Öktem, Inverse Problems '17.

Example: Learned iterative unrolling

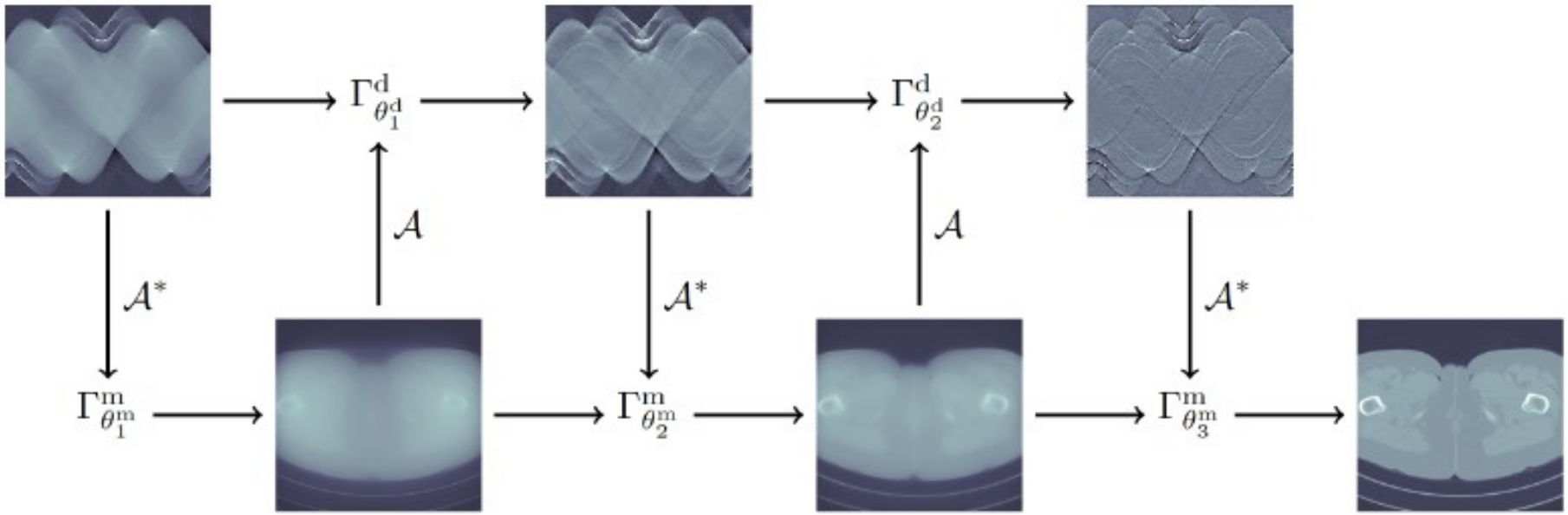
Algorithm 1 Non-linear primal dual hybrid gradient

Given: $\sigma, \tau > 0$ s.t. $\sigma\tau\|\mathcal{K}\|^2 < 1$, $\gamma \in [0,1]$ and $f_0 \in X$, $h_0 \in U$.
for $i = 1, \ldots$ **do**
  $h_{i+1} \leftarrow \text{prox}_{\sigma\mathcal{F}^*}\left(h_i + \sigma\,\mathcal{K}(\bar{f}_i)\right)$
  $f_{i+1} \leftarrow \text{prox}_{\tau\mathcal{G}}\left(f_i - \tau[\partial\,\mathcal{K}(f_i)]^*(h_{i+1})\right)$
  $\bar{f}_{i+1} \leftarrow f_{i+1} + \gamma(f_{i+1} - f_i)$

$$\text{prox}_{\tau\mathcal{G}}(f) = \arg\min_{f' \in X}\left[\mathcal{G}(f') + \frac{1}{2\tau}\|f' - f\|_X^2\right]$$

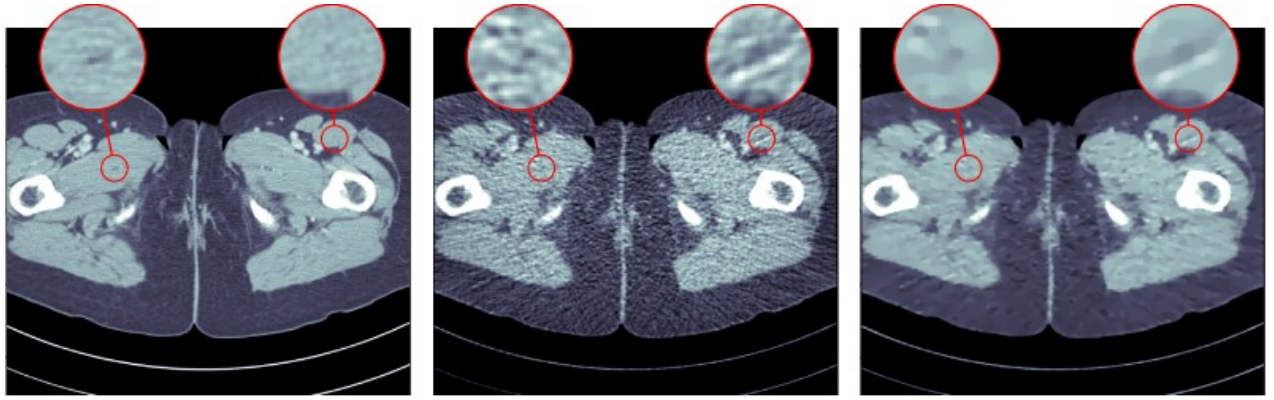
Adler, Jonas, and Ozan Öktem. "Learned primal-dual reconstruction." *IEEE transactions on medical imaging* 37.6 (2018): 1322-1332.

Example: Learned Primal Dual



Adler, Öktem, Inverse Problems '17

Example: Learned Primal Dual



(a) 512 × 512 pixel human phantom

(b) Filtered back-projection (FBP)
PSNR 33.65 dB, SSIM 0.830, 423 ms

(c) Total variation (TV)
PSNR 37.48 dB, SSIM 0.946, 64 371 ms

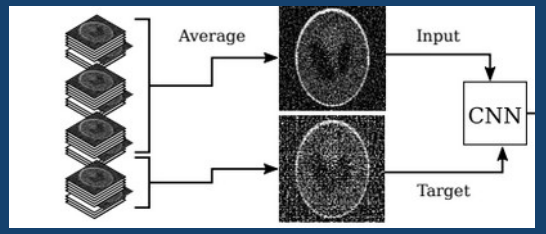
(d) FBP + U-Net denoising
PSNR 41.92 dB, SSIM 0.941, 463 ms

(e) Primal-Dual, linear
PSNR 44.10 dB, SSIM 0.969, 620 ms

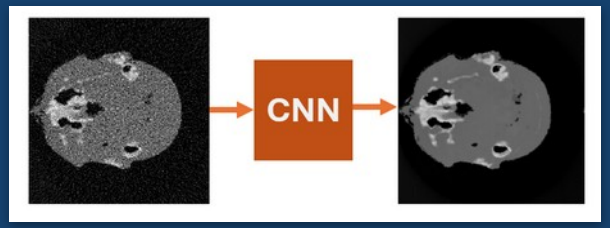
(f) Primal-Dual, non-linear
PSNR 43.91 dB, SSIM 0.969, 670 ms

Data Driven Tomographic Reconstruction

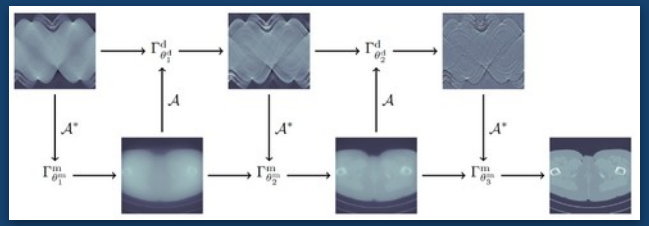
Self/Unsupervised



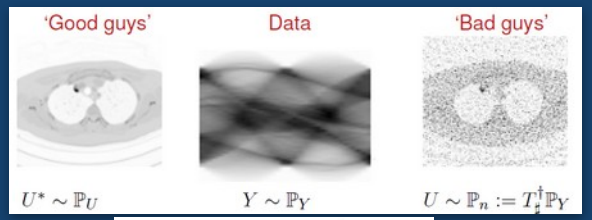
Post-Processing



Unrolled Iterative Methods



Learning a regulariser



$$x_{reg} = \arg\min_x \|Ax - b\|_2^2 + \lambda R(x)$$

Plug and Play

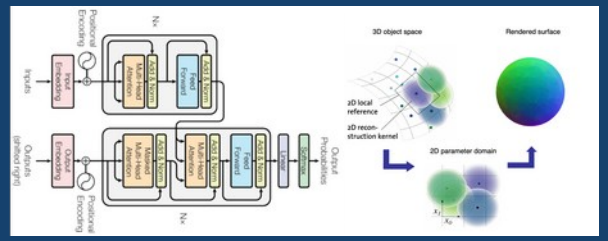
$$\mathbf{x}^{k+1} = \operatorname*{argmin}_{\mathbf{x}} D(\mathbf{A}\mathbf{x}, \mathbf{b}) + \lambda \|\mathbf{x} - \mathbf{v}^k + \mathbf{d}^k\|$$

Learned Denoiser

$$\mathbf{v}^{k+1} = \operatorname*{argmin}_{\mathbf{v}} R(\mathbf{v}) + \lambda \|\mathbf{v} - \mathbf{x}^{k+1} + \mathbf{d}^k\|$$

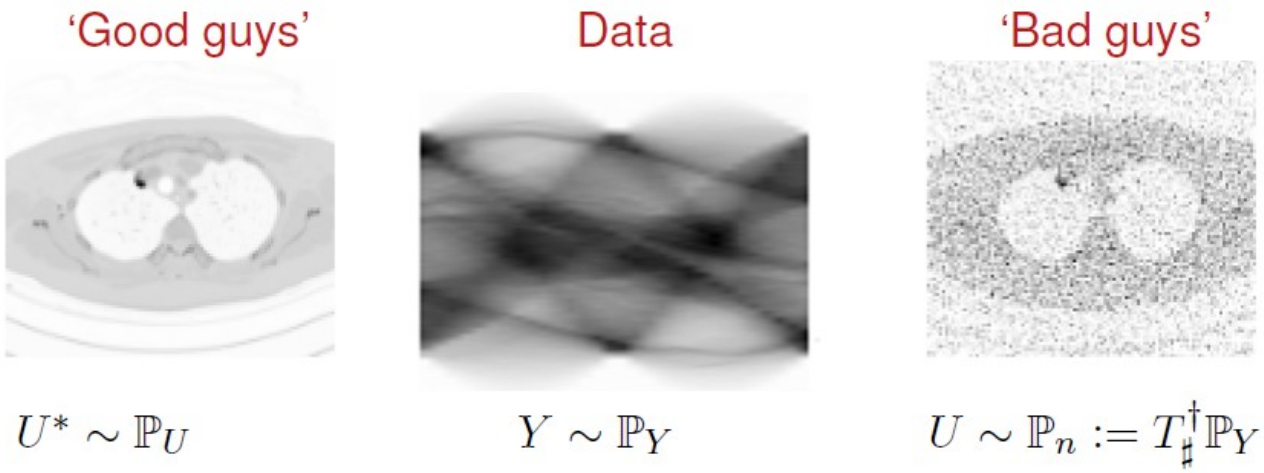
$$\mathbf{d}^{k+1} = \mathbf{d}^k - (\mathbf{x}^{k+1} - \mathbf{v}^{k+1})$$

Other ML methods



Learned Regularization: Adversarial regularizer

$$\hat{\mu} = \underset{\mu}{\operatorname{argmin}}\ L$$



Train regulariser in adversarial manner to be small for samples from $\mathbb{P}_U$ and large for samples from $\mathbb{P}_n$.

Lunz, Öktem, CBS, NeurIPS '18; Arridge, Maass, Öktem, CBS, Acta Numerica '19; Lunz, PhD thesis University of Cambridge '21

Learned Regularization

- Idea from Wasserstein GANs [16]: use 1-Wasserstein distance as (unsupervised) loss for regulariser

$$\mathrm{Wass}(\mathbb{P}_n, \mathbb{P}_U) = \sup_{R \in 1-Lip} \mathcal{E}_{U\sim\mathbb{P}_n} R(U) - \mathcal{E}_{U\sim\mathbb{P}_U} R(U)$$

- Train the regulariser to learn image statistics – good and bad guys DO NOT need to be paired :)
- Parametrization for R^*: $R_\Theta(u) = \Psi_\Theta(u) + \rho_0 \|u\|_2^2$, where $\Psi_\Theta(u)$ is a [convex[17]] CNN.
- Train the Network with the loss [18]

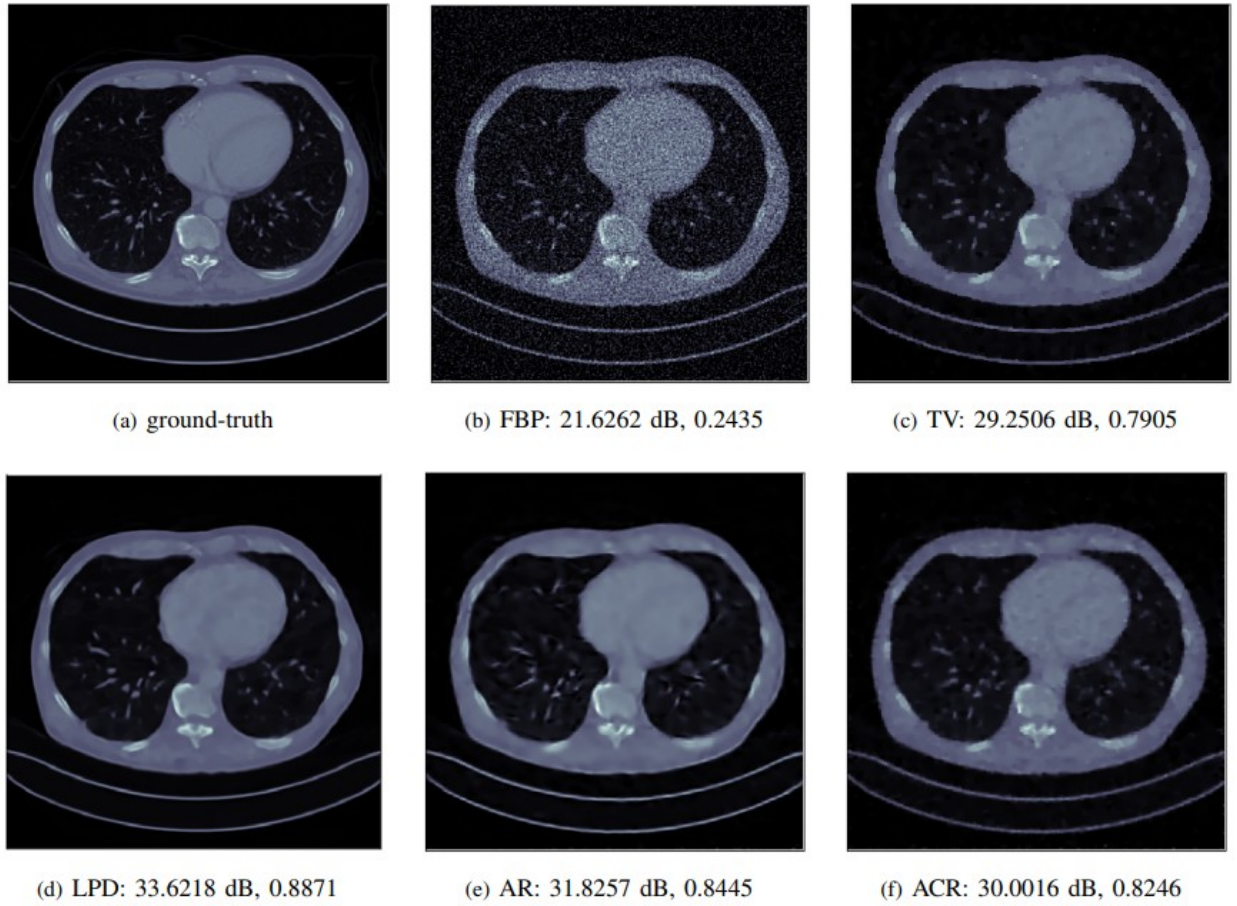
$$\min_\Theta \mathbb{E}_{U\sim\mathbb{P}_U}[\Psi_\Theta(U)] - \mathbb{E}_{U\sim\mathbb{P}_n}[\Psi_\Theta(U)] + \mu \cdot \mathbb{E}\left[(\|\nabla_u \Psi_\Theta(U)\|_* - 1)_+^2\right].$$

[16] Arjovsky, Chintala & Bottou, '17

[17] Amos, Xu, Kolter ICML '17; S. Mukherjee, S. Dittmer, Z. Shumaylov, S. Lunz, O. Öktem, CBS, arXiv:2008.02839

[18] Gulrajani et al., '17

Example: Adversarial Convex Regularizer

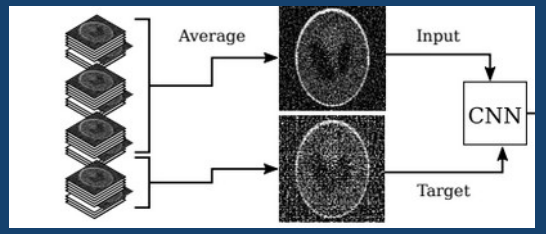


(a) ground-truth (b) FBP: 21.6262 dB, 0.2435 (c) TV: 29.2506 dB, 0.7905

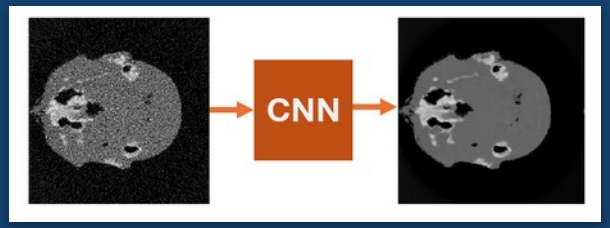
(d) LPD: 33.6218 dB, 0.8871 (e) AR: 31.8257 dB, 0.8445 (f) ACR: 30.0016 dB, 0.8246

Data Driven Tomographic Reconstruction

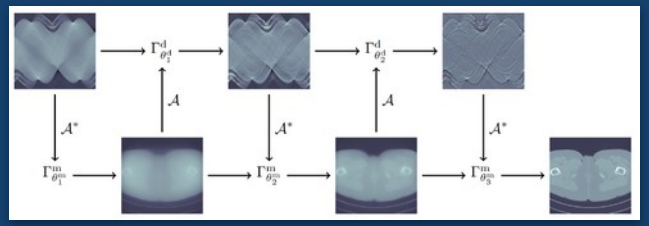
Self/Unsupervised



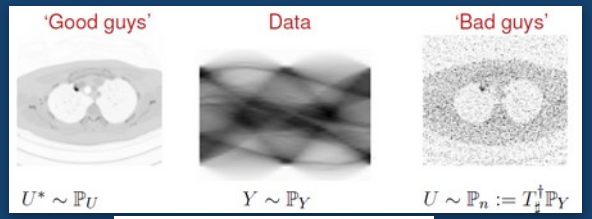
Post-Processing



Unrolled Iterative Methods



Learning a regulariser



$$x_{reg} = \arg\min_x \|Ax - b\|_2^2 + \lambda R(x)$$

Plug and Play

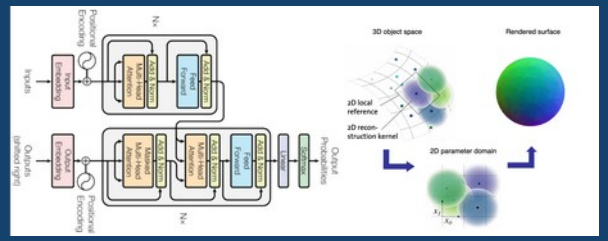
$$\mathbf{x}^{k+1} = \operatorname{argmin}_{\mathbf{x}} D(\mathbf{Ax}, \mathbf{b}) + \lambda \|\mathbf{x} - \mathbf{v}^k + \mathbf{d}^k\|$$

Learned Denoiser

$$\mathbf{v}^{k+1} = \operatorname{argmin}_{\mathbf{v}} R(\mathbf{v}) + \lambda \|\mathbf{v} - \mathbf{x}^{k+1} + \mathbf{d}^k\|$$

$$\mathbf{d}^{k+1} = \mathbf{d}^k - (\mathbf{x}^{k+1} - \mathbf{v}^{k+1})$$

Other ML methods



Plug and Play

$$\hat{\mathbf{x}} = \underset{\mathbf{x}}{\operatorname{argmin}}\ L$$

Can be solved with e.g. ADMM as:

$$\mathbf{x}^{k+1} = \underset{\mathbf{x}}{\operatorname{argmin}}\ D(\mathbf{A}\mathbf{x}, \mathbf{b}) + \lambda \left\| \mathbf{x} - \mathbf{v}^k + \mathbf{d}^k \right\|$$

$$\mathbf{v}^{k+1} = \underset{\mathbf{v}}{\operatorname{argmin}}\ R(\mathbf{v}) + \lambda \left\| \mathbf{v} - \mathbf{x}^{k+1} + \mathbf{d}^k \right\|$$

$$\mathbf{d}^{k+1} = \mathbf{d}^k - (\mathbf{x}^{k+1} - \mathbf{v}^{k+1})$$

Denoiser !

Venkatakrishnan, Bouman, Wohlberg, IEEE SIP '13

Plug and Play

- STEP 1: Train a denoiser (like the post-processing networks)

- STEP 2: Use the denoiser when you reconstruct a new image

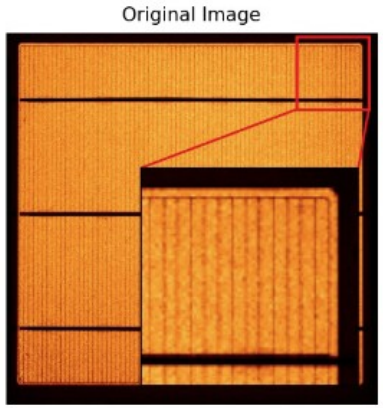
$$\mathbf{x}^{k+1} = \underset{\mathbf{x}}{\operatorname{argmin}}\ D(\mathbf{A}\mathbf{x}, \mathbf{b}) + \lambda \left\| \mathbf{x} - \mathbf{v}^{k} + \mathbf{d}^{k} \right\|$$

$$\mathbf{v}^{k+1} = \Gamma_{\theta}(\mathbf{v} - \mathbf{x}^{k+1} + \mathbf{d}^{k})$$

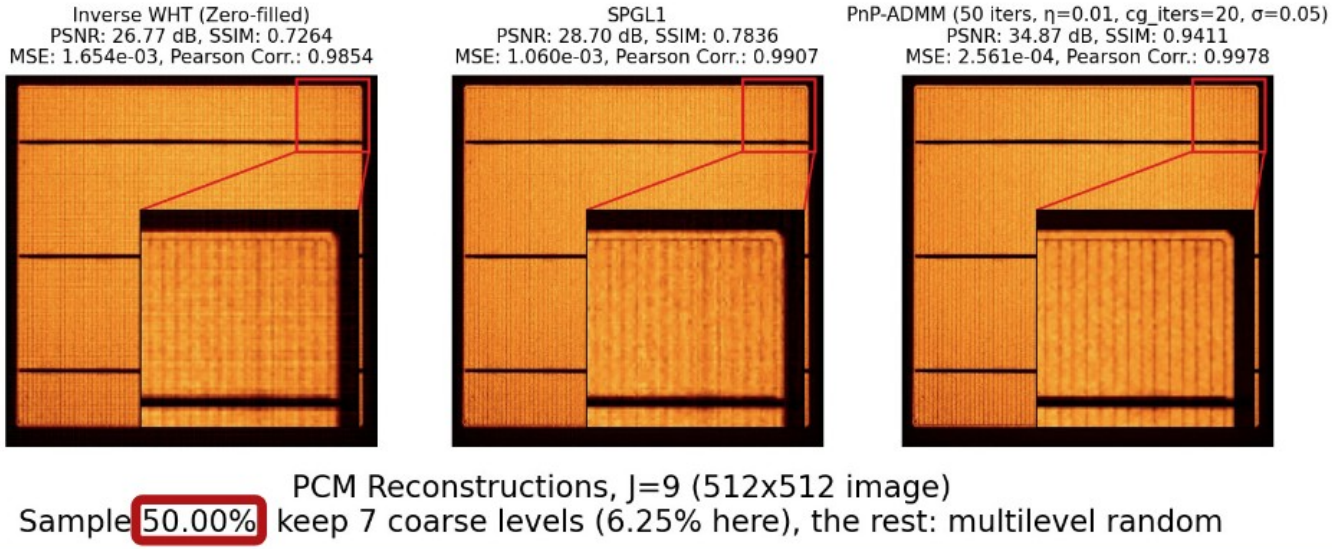
$$\mathbf{d}^{k+1} = \mathbf{d}^{k} - (\mathbf{x}^{k+1} - \mathbf{v}^{k+1})$$

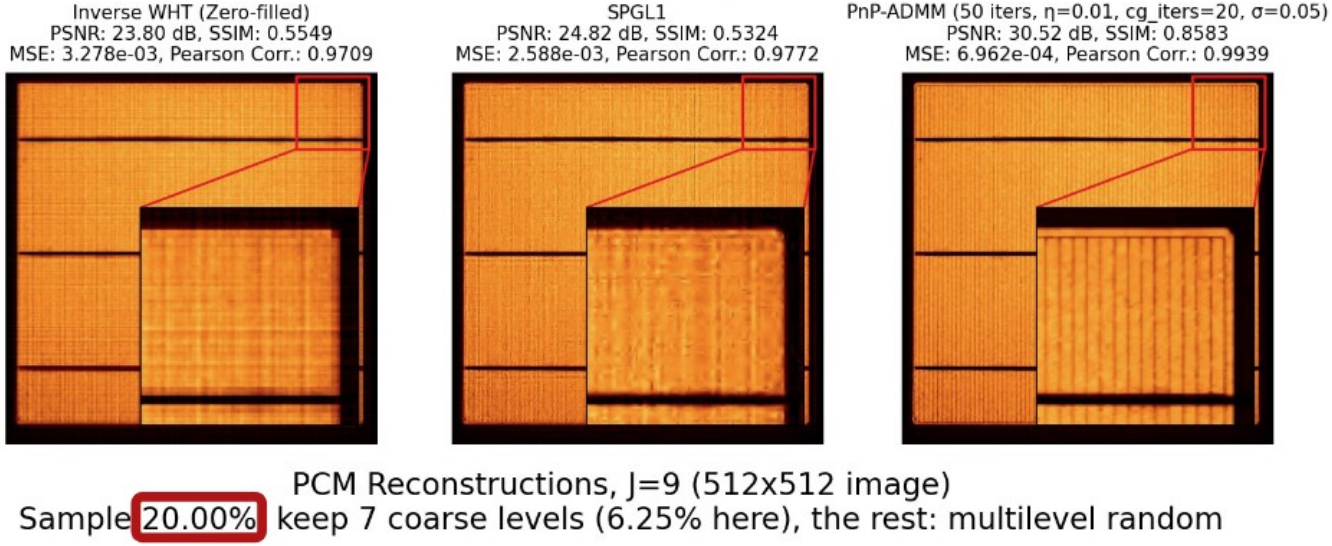
$\mathbf{v}^{k+1} = \Gamma_{\theta}(\mathbf{v} - \mathbf{x}^{k+1} + \mathbf{d}^{k})$

Example:



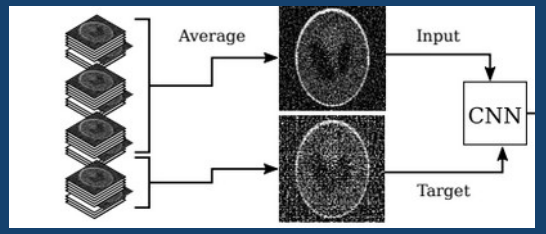
Silicon sample (512 x 512) (normalized pixels in [0, 1])



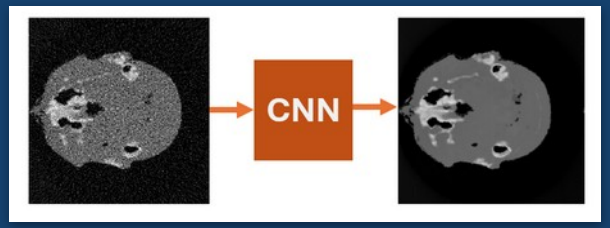


Data Driven Tomographic Reconstruction

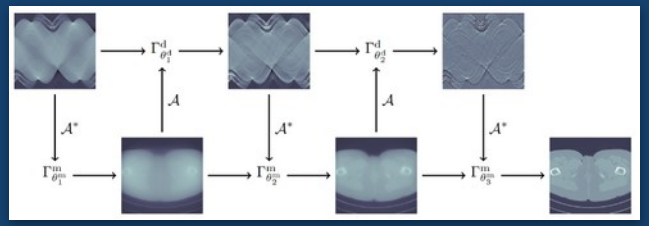
Self/Unsupervised



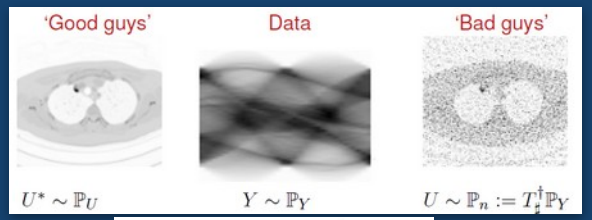
Post-Processing



Unrolled Iterative Methods



Learning a regulariser



$$x_{reg} = \arg\min_x \|Ax - b\|_2^2 + \lambda R(x)$$

Plug and Play

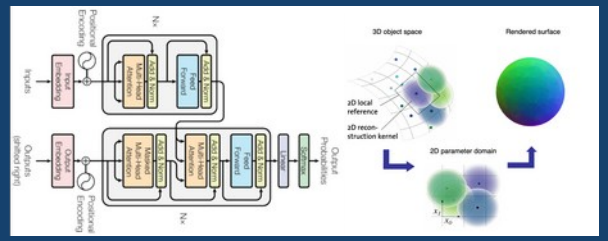
$$\mathbf{x}^{k+1} = \operatorname*{argmin}_{\mathbf{x}} D(\mathbf{A}\mathbf{x}, \mathbf{b}) + \lambda \|\mathbf{x} - \mathbf{v}^k + \mathbf{d}^k\|$$

Learned Denoiser

$$\mathbf{v}^{k+1} = \operatorname*{argmin}_{\mathbf{v}} R(\mathbf{v}) + \lambda \|\mathbf{v} - \mathbf{x}^{k+1} + \mathbf{d}^k\|$$

$$\mathbf{d}^{k+1} = \mathbf{d}^k - (\mathbf{x}^{k+1} - \mathbf{v}^{k+1})$$

Other ML methods



Unsupervised learning



Alvaro **Esposito**

- Example: Noise2Inverse
  - By splitting the problem in worse reconstruction, you can learn to denoise!

