# Mathematics of X-ray Computed Tomography: Deep Learning and Learned Reconstruction Methods

**Tatiana A. Bubba**

Department of Mathematics and Computer Science, University of Ferrara

tatiana.bubba@unife.it

**Physical Sciences Summer School**

Puerto Varas, 5-8 January 2026

UNIVERSITÀ DEGLI STUDI DI FERRARA
- EX LABORE FRUCTUS -

MiVA

European Women in Mathematics

# Brief Recap: Regularization of (Tomographic) Inverse Problem

Recall the (tomographic) inverse problem:

$$\text{given noisy measurements } y^{\delta} = \mathcal{A}f, \text{ determine } \; f$$

where $\mathcal{A} : X \to Y$ (e.g., $\mathcal{A} = \mathcal{R}$ Radon transform), $f \in X$ is the unknown (parameters), $y^{\delta} = y + \epsilon \in Y$ is the noisy measurements with $\epsilon$ s.t. $\|\epsilon\| \leq \delta$, measurement noise.

# Brief Recap: Regularization of (Tomographic) Inverse Problem

Recall the (tomographic) inverse problem:

$$\text{given noisy measurements } y^{\delta} = \mathcal{A}f, \text{ determine } \; f$$

where $\mathcal{A} : X \to Y$ (e.g., $\mathcal{A} = \mathcal{R}$ Radon transform), $f \in X$ is the unknown (parameters), $y^{\delta} = y + \epsilon \in Y$ is the noisy measurements with $\epsilon$ s.t. $\|\epsilon\| \leq \delta$, measurement noise.

Variational regularization :

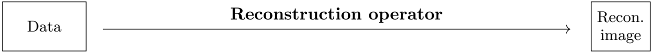
$$\operatorname*{argmin}_{\boldsymbol{f} \in X} \left\{ \frac{1}{2} \left\| \mathcal{A}f - y^{\delta} \right\|_Y^2 + \alpha \mathsf{Reg}(f) \right\}$$

Key components:

- $X$ = possible signals, $Y$ = possible data
- Data model: How data is generated (forward model or simulator)
- Prior model: Characteristics of “natural” signals (regularisation)

# Model-based Approaches in Inverse Problems

**Analytical approaches:** explicit reconstruction operator for direct inversion



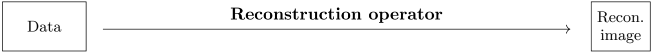
Data

**Reconstruction operator**

Recon. image

# Model-based Approaches in Inverse Problems

**Analytical approaches:** explicit reconstruction operator for direct inversion



Data

**Reconstruction operator**

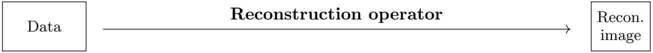
Recon. image

**Model-based approaches:** the reconstruction operator is given implicitly (loop)

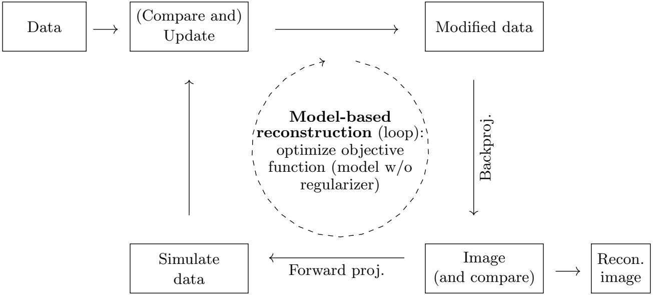
1. Generate simulated data from a signal
2. Measure mismatch:
    - Simulated data against observed data
    - Signal against prior model
3. Update signal bearing in mind mismatch
4. Repeat until some stopping criteria is fulfilled

# Model-based Approaches in Inverse Problems

**Analytical approaches:** explicit reconstruction operator for direct inversion



**Model-based approaches:** the reconstruction operator is given implicitly (loop)



# Model-based vs Data-driven Approaches

**Challenges** of model-based classical approaches:

➢ Hand-tuning of parameters: such as the regularization parameter in variational methods (or the frequency scaling in FBP for tomography)

➢ Choice and design of regularizer/constructing good prior models: handcrafted regularizer tend to encode simplistic features of the true solution

➢ Computational feasibility/scalability: state-of-the-art performance is often computationally heavy (e.g., solution via high-dimensional optimisation problem with repeated application of the forward operator and adjoint)

# Model-based vs Data-driven Approaches

**Opportunities** of data-driven approaches:

- Hand-tuning of parameters: such as the regularization parameter in variational methods (or the frequency scaling in FBP for tomography)

  Learned approaches find optimal parameters from training data, or a data dependent parameter choice rule

- Choice and design of regularizer/constructing good prior models: handcrafted regularizer tend to encode simplistic features of the true solution

  Learned approaches offer a flexible framework to learn suitable regularizer from a collection of representative examples (training data)

- Computational feasibility/scalability: state-of-the-art performance is often computationally heavy (e.g., solution via high-dimensional optimisation problem with repeated application of the forward operator and adjoint)

  Learned approaches offer a wide range of solutions, from improving simple reconstructions to learning more efficient update rules in iterative methods

# Model-based vs Data-driven Approaches

**Opportunities** of data-driven approaches:

- Hand-tuning of parameters: such as the regularization parameter in variational methods (or the frequency scaling in FBP for tomography)

  Learned approaches find optimal parameters from training data, or a data dependent parameter choice rule

- Choice and design of regularizer/constructing good prior models: handcrafted regularizer tend to encode simplistic features of the true solution

  Learned approaches offer a flexible framework to learn suitable regularizer from a collection of representative examples (training data)

- Computational feasibility/scalability: state-of-the-art performance is often computationally heavy (e.g., solution via high-dimensional optimisation problem with repeated application of the forward operator and adjoint)

  Learned approaches offer a wide range of solutions, from improving simple reconstructions to learning more efficient update rules in iterative methods

**Learned reconstructions:** Combine best of both worlds (model-based and data-driven approaches) for solving inverse problems

# Defining the Learning Task

Goal: Learning a reconstruction operator

- We aim to solve the inverse problem $\mathcal{A}f = y$, given $y^\delta \in Y$
- (Many) inverse problems are ill-posed/ill-conditioned
  $\rightsquigarrow$ Rather learn a regularized map, which is well-posed
- We want to learn a reconstruction operator $\mathscr{R} : Y \to X$ which may consist of learned and model-based parts

# Defining the Learning Task

Goal: Learning a reconstruction operator

- We aim to solve the inverse problem $\mathcal{A}f = y$, given $y^\delta \in Y$
- (Many) inverse problems are ill-posed/ill-conditioned
  $\rightsquigarrow$ Rather learn a regularized map, which is well-posed
- We want to learn a reconstruction operator $\mathscr{R} : Y \to X$ which may consist of learned and model-based parts

We need to slightly reformulate our inverse problem, we consider:

$$\mathcal{A}f + \epsilon = y$$

where we assume that:

- Signals $f$ in $X$ are generated by a $X$-valued random variable $\mathtt{f}$
- $\mathtt{y}$ is a $Y$-valued random variable whose samples $y$ represent possible data
- Observed data $y \in Y$ are a single sample of the $Y$-valued conditional random variable $(\mathtt{y}|\mathtt{f} = f^{\mathrm{gt}})$ where $f^{\mathrm{gt}} \in X$ is the (unknown) true signal

# Defining the Learning Task

Goal: Learning a reconstruction operator

- We aim to solve the inverse problem $\mathcal{A}f = y$, given $y^\delta \in Y$
- (Many) inverse problems are ill-posed/ill-conditioned
  $\rightsquigarrow$ Rather learn a regularized map, which is well-posed
- We want to learn a reconstruction operator $\mathscr{R} : Y \to X$ which may consist of learned and model-based parts

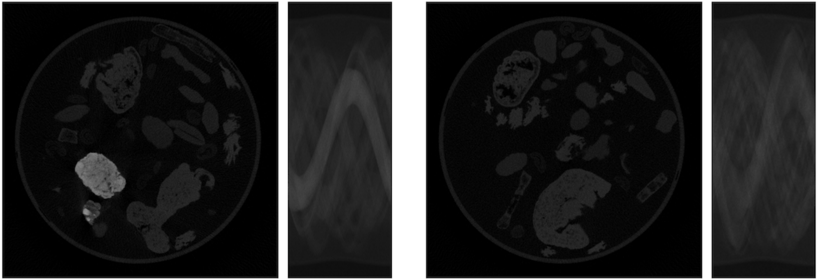
## Checklist

What do we need to assess?

- Type of training data
- Choice of loss function
- Learning task: formulating (parametrizing) the reconstruction operator

# Training data: Supervised

We have paired training data in $X \times Y$ that are independent samples of $(\mathtt{f}, \mathtt{y})$ where $\mathcal{A}f + \epsilon = y$ holds.

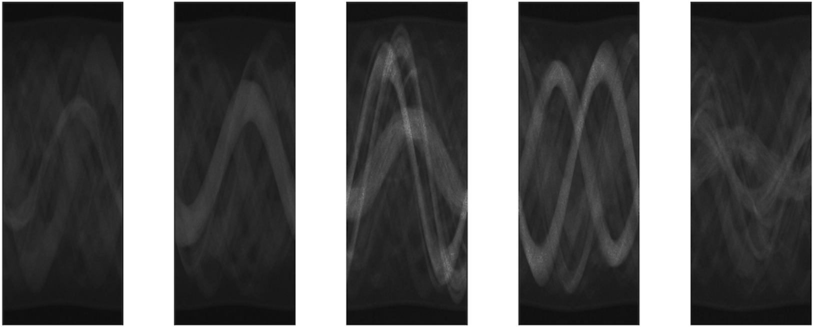


**Example:** matching pairs of low-dose data and high-dose reconstruction

[Image credits: M. Kiss et al., Scientific Data, 2023]

# Training data: Unsupervised

In this case, training data are independent samples $y_1, \ldots, y_n$ in $Y$ of $\mathrm{y}$.



**Example:** samples of low-dose measurements

[Image credits: M. Kiss et al., Scientific Data, 2023]

# Loss Functions: Supervised Setting

Things to keep in mind in general:

- Type of training data dictates possible loss functions
- Loss functions can be formulated in $X$ or $Y$

# Loss Functions: Supervised Setting

Things to keep in mind in general:

- Type of training data dictates possible loss functions
- Loss functions can be formulated in $X$ or $Y$

## Supervised setting

Training data are i.i.d. samples $(f_1, y_1), \dots, (f_n, y_n) \in X \times Y$.

- The most common setting is with $X$-space loss $\ell_X : X \times X \to \mathbb{R}$, e.g., $p$-norm. The learned reconstruction operator $\mathscr{R}_\theta$ is the associated Bayes estimator, i.e., $\mathscr{R}_{\widehat{\theta}} : Y \to X$ where $\widehat{\theta} \in \Theta$ solves:

$$\widehat{\theta} \in \operatorname*{argmin}_{\theta \in \Theta} \mathbb{E}_{\mathtt{f},\mathtt{y}} \left[ \ell_X(\mathscr{R}_\theta(y), f) \right]$$

**Example:** $\ell_X = \| \cdot \|_2^2 \quad \Rightarrow \quad \mathscr{R}_{\widehat{\theta}} =$ posterior mean.

# Loss Functions: Supervised Setting

Things to keep in mind in general:

- Type of training data dictates possible loss functions
- Loss functions can be formulated in $X$ or $Y$

## Supervised setting

Training data are i.i.d. samples $(f_1, y_1), \ldots, (f_n, y_n) \in X \times Y$.

- The most common setting is with $X$-space loss $\ell_X : X \times X \to \mathbb{R}$, e.g., $p$-norm. The learned reconstruction operator $\mathscr{R}_\theta$ is the associated Bayes estimator, i.e., $\mathscr{R}_{\widehat{\theta}} : Y \to X$ where $\widehat{\theta} \in \Theta$ solves:

$$\widehat{\theta} \in \operatorname*{argmin}_{\theta \in \Theta} \mathbb{E}_{\mathrm{f},\mathrm{y}} \left[ \ell_X(\mathscr{R}_\theta(y), f) \right]$$

**Example:** $\ell_X = \| \cdot \|_2^2 \quad \Rightarrow \quad \mathscr{R}_{\widehat{\theta}} =$ posterior mean.

- In practice, we compute the corresponding empirical risk estimator:

$$\widehat{\theta} \in \operatorname*{argmin}_{\theta \in \Theta} \frac{1}{n} \sum_{i=1}^{n} \ell_X(\mathscr{R}_\theta(y_i), f_i)$$

# Loss Functions: Unsupervised Setting

The training data are independent samples of $\mathrm{y}$, i.e., $y_1, \ldots, y_n \in Y$.

- ➤ Without using any further information, we can only formulate a loss in $X$ that measures the goodness of the reconstruction with a suitable loss $\ell_X : X \to \mathbb{R}$, i.e., we aim to find $\mathscr{R}_\theta$ such that

$$\widehat{\theta} \in \operatorname*{argmin}_{\theta \in \Theta} \mathbb{E}_{\mathrm{y}} \left[ \ell_X(\mathscr{R}_\theta(y)) \right]$$

Different strategies are devised by introducing more structure into the learning problem.

# Loss Functions: Unsupervised Setting

The training data are independent samples of $\mathrm{y}$, i.e., $y_1, \ldots, y_n \in Y$.

- ➤ Without using any further information, we can only formulate a loss in $X$ that measures the goodness of the reconstruction with a suitable loss $\ell_X : X \to \mathbb{R}$, i.e., we aim to find $\mathscr{R}_\theta$ such that

$$\widehat{\theta} \in \operatorname*{argmin}_{\theta \in \Theta} \mathbb{E}_{\mathrm{y}} \left[ \ell_X (\mathscr{R}_\theta(y)) \right]$$

  Different strategies are devised by introducing more structure into the learning problem.

- ➤ Alternative #1: Mimic the supervised setting by creating a reference reconstruction $\mathscr{R}(y)$, where $\mathscr{R}$ may be a gold-standard algorithm:

$$\widehat{\theta} \in \operatorname*{argmin}_{\theta \in \Theta} \mathbb{E}_{\mathrm{y}} \left[ \ell_X (\mathscr{R}_\theta(y), \mathscr{R}(y)) \right]$$

  When $\mathscr{R}$ solves an optimisation problem, this corresponds to the paradigm of learning to optimise.

# Loss Functions: Unsupervised Setting

The training data are independent samples of $\mathrm{y}$, i.e., $y_1, \dots, y_n \in Y$.

- ➤ Without using any further information, we can only formulate a loss in $X$ that measures the goodness of the reconstruction with a suitable loss $\ell_X : X \to \mathbb{R}$, i.e., we aim to find $\mathscr{R}_\theta$ such that

  $$\widehat{\theta} \in \operatorname*{argmin}_{\theta \in \Theta} \mathbb{E}_{\mathrm{y}} \left[ \ell_X(\mathscr{R}_\theta(y)) \right]$$

  Different strategies are devised by introducing more structure into the learning problem.

- ➤ Alternative #2: Split the data space disjointly into $Y = Y_1 \cup Y_2$ and then consider training data that are paired random samples from $Y_1$ and $Y_2$-valued random variables $\mathrm{y}_1$ and $\mathrm{y}_2$, resp.:

  $$\widehat{\theta} \in \operatorname*{argmin}_{\theta \in \Theta} \mathbb{E}_{\mathrm{y}} \left[ \ell_X(\mathscr{R}_\theta(y_1), \mathscr{R}(y_2)) \right]$$

  This allows exploitation of the structure of $\mathcal{A}$, and hence that of $\mathscr{R}$, and the data itself. It is often referred to as self-supervised (e.g., Noise2Noise).

# Loss Functions: Unsupervised Setting

The training data are independent samples of $\mathrm{y}$, i.e., $y_1, \ldots, y_n \in Y$.

- Without using any further information, we can only formulate a loss in $X$ that measures the goodness of the reconstruction with a suitable loss $\ell_X : X \to \mathbb{R}$, i.e., we aim to find $\mathscr{R}_\theta$ such that

  $$\widehat{\theta} \in \operatorname*{argmin}_{\theta \in \Theta} \mathbb{E}_{\mathrm{y}} \left[ \ell_X(\mathscr{R}_\theta(y)) \right]$$

  Different strategies are devised by introducing more structure into the learning problem.

- Alternative #3: Measure goodness in data space $Y$ so that we do not require a handcrafted reconstruction operator $\mathscr{R}$, but need $\mathcal{A}$ for projection back into $Y$:

  $$\widehat{\theta} \in \operatorname*{argmin}_{\theta \in \Theta} \mathbb{E}_{\mathrm{y}} \left[ \ell_Y((\mathcal{A} \circ \mathscr{R}_\theta(y), y) \right]$$

  When $\ell_Y$ is the $\ell^2$-norm, this amounts to the least-squares problem, which can lead to overfitting without further regularization.

  **Variant.** Consider a single $y \in Y$ and then learn a network $\Lambda_\theta$ as generator of $f$: Deep Image Prior (DIP).

# Formulating a Suitable Reconstruction Operator

Main considerations influencing the choice of parameterisation for $\mathscr{R}_\theta : Y \to X$:

- Amount of training data
- Computational resources
- Forward and adjoint availability
- Quantitative performance
- Theoretical guarantees and interpretability

# Formulating a Suitable Reconstruction Operator

Main considerations influencing the choice of parameterisation for $\mathscr{R}_\theta : Y \to X$:

- Amount of training data
- Computational resources
- Forward and adjoint availability
- Quantitative performance
- Theoretical guarantees and interpretability

Two primary paradigms for learning a reconstruction operator:

- Uncoupled: The training process (of the network) is decoupled from the model for generating the inverse problem data, i.e., training does not involve evaluating the forward operator or the adjoint.
- Learned iterative schemes: The network components and forward/adjoint are intertwined, i.e., training the reconstructions operator necessarily requires evaluation of the model.

# Learned Reconstruction Operator

## Definition (learned reconstruction operator)

A family of parametrised mappings

$$\mathscr{R}_\theta : Y \to X \qquad \text{where} \;\; \theta \in \Theta$$

is called a learned reconstruction operator for the inverse problem $\mathcal{A}f = y$ if the parameters $\theta$ are determined (learned) from example data (training data) that is generated in a way that is consistent with $\mathcal{A}f = y$.

# Learned Reconstruction Operator

### Definition (learned reconstruction operator)

A family of parametrised mappings

$$\mathscr{R}_\theta : Y \to X \qquad \text{where} \;\; \theta \in \Theta$$

is called a learned reconstruction operator for the inverse problem $\mathcal{A}f = y$ if the parameters $\theta$ are determined (learned) from example data (training data) that is generated in a way that is consistent with $\mathcal{A}f = y$.

**Examples:**

1. Two-step approaches:

$$\mathscr{R}_\theta = \Lambda_\theta \circ \mathscr{R}, \quad \text{with} \;\; \Lambda_\theta : X \to X \qquad \text{(post-processing)}$$

$$\mathscr{R}_\theta = \mathscr{R} \circ \Lambda_\theta, \quad \text{with} \;\; \Lambda_\theta : Y \to Y \qquad \text{(pre-processing)}$$

where $\mathscr{R}$ is a hand-crafted reconstruction operator.

# Learned Reconstruction Operator

**Definition (learned reconstruction operator)**

A family of parametrised mappings

$$\mathscr{R}_\theta : Y \to X \qquad \text{where} \quad \theta \in \Theta$$

is called a learned reconstruction operator for the inverse problem $\mathcal{A}f = y$ if the parameters $\theta$ are determined (learned) from example data (training data) that is generated in a way that is consistent with $\mathcal{A}f = y$.

**Examples:**

1. Two-step approaches:

$$\mathscr{R}_\theta = \Lambda_\theta \circ \mathscr{R}, \quad \text{with} \;\; \Lambda_\theta : X \to X \qquad \text{(post-processing)}$$

$$\mathscr{R}_\theta = \mathscr{R} \circ \Lambda_\theta, \quad \text{with} \;\; \Lambda_\theta : Y \to Y \qquad \text{(pre-processing)}$$

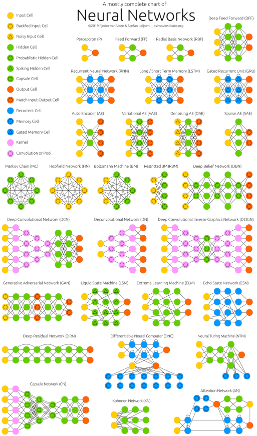
   where $\mathscr{R}$ is a hand-crafted reconstruction operator.

2. Learned regulariser: Define $\Lambda_\theta : X \to \mathbb{R}$, then:

$$\mathscr{R}_\theta(y) = \operatorname*{argmin}_{f \in X} D(f) + \Lambda_\theta(f).$$

# Neural Network Architectures

Architecture: precise description of how $\theta$ parametrises $\Lambda_\theta$.



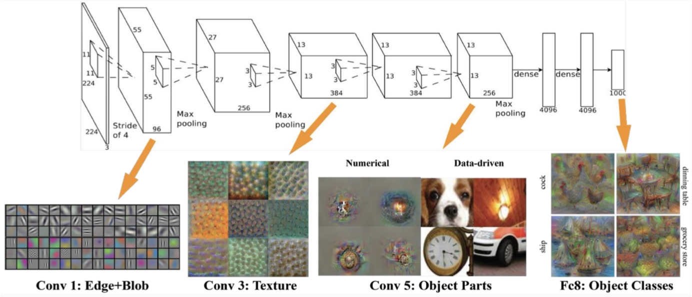
A mostly complete chart of
Neural Networks
©2019 Fjodor van Veen & Stefan Leijnen asimovinstitute.org
Input Cell
Backfed Input Cell
Noisy Input Cell
Hidden Cell
Probablistic Hidden Cell
Spiking Hidden Cell
Capsule Cell
Output Cell
Match Input Output Cell
Recurrent Cell
Memory Cell
Gated Memory Cell
Kernel
Convolution or Pool
Perceptron (P)
Feed Forward (FF)
Radial Basis Network (RBF)
Deep Feed Forward (DFF)
Recurrent Neural Network (RNN)
Long / Short Term Memory (LSTM)
Gated Recurrent Unit (GRU)
Auto Encoder (AE)
Variational AE (VAE)
Denoising AE (DAE)
Sparse AE (SAE)
Markov Chain (MC)
Hopfield Network (HN)
Boltzmann Machine (BM)
Restricted BM (RBM)
Deep Belief Network (DBN)
Deep Convolutional Network (DCN)
Deconvolutional Network (DN)
Deep Convolutional Inverse Graphics Network (DCIGN)
Generative Adversarial Network (GAN)
Liquid State Machine (LSM)
Extreme Learning Machine (ELM)
Echo State Network (ESN)
Deep Residual Network (DRN)
Differentiable Neural Computer (DNC)
Neural Turing Machine (NTM)
Capsule Network (CN)
Kohonen Network (KN)
Attention Network (AN)

**Examples:**

- Single layer network
- Sequential deep network
- Convolutional neural networks (CNNs)
- Unrolled/Unfolded neural networks
- Recurrent neural networks (RNNs)

# Convolutional Neural Networks

CNN = NN with convolution operation instead of matrix multiplication



**Core idea:** use geometry of data (proximity, directions)

- Suitable for imaging problems (*e.g.*, feature extraction)
- Very good performance for (approximately) translation invariant tasks
- Sparse connectivity, parameter sharing, many layers

# A Very Famous CNN: U-Net

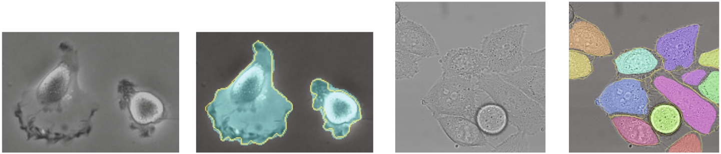
O. Ronneberger, P. Fischer and T. Brox
U-Net: Convolutional Networks for Biomedical Image Segmentation
MICCAI 2015: 18th International Conference, 234-241.

# A Very Famous CNN: U-Net

O. Ronneberger, P. Fischer and T. Brox
U-Net: Convolutional Networks for Biomedical Image Segmentation
MICCAI 2015: 18th International Conference, 234-241.

The U-Net architecture is one of the most important and foundational neural network architectures of today.

- U-Net was initially applied to the segmentation of medical images.



# A Very Famous CNN: U-Net

O. Ronneberger, P. Fischer and T. Brox
U-Net: Convolutional Networks for Biomedical Image Segmentation
MICCAI 2015: 18th International Conference, 234-241.

The U-Net architecture is one of the most important and foundational neural network architectures of today.

- U-Net was initially applied to the segmentation of medical images.
- It turned out to be useful for many more computer vision tasks, with semantic segmentation applications being a prominent example.

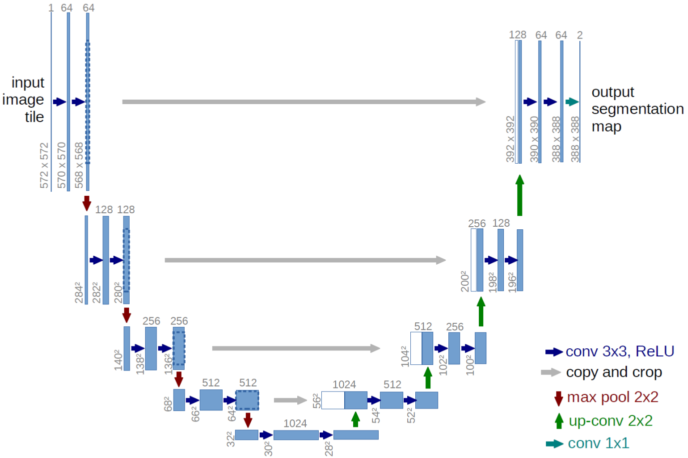
# A Very Famous CNN: U-Net

O. Ronneberger, P. Fischer and T. Brox
U-Net: Convolutional Networks for Biomedical Image Segmentation
MICCAI 2015: 18th International Conference, 234-241.

The U-Net architecture is one of the most important and foundational neural network architectures of today.

- U-Net was initially applied to the segmentation of medical images.
- It turned out to be useful for many more computer vision tasks, with semantic segmentation applications being a prominent example.
- Many variants of the U-net architecture have been developed and applied in many diverse imaging tasks, including in diffusion models for image generation models, such as DALL-E and Midjourney.

# U-Net: The Original Architecture



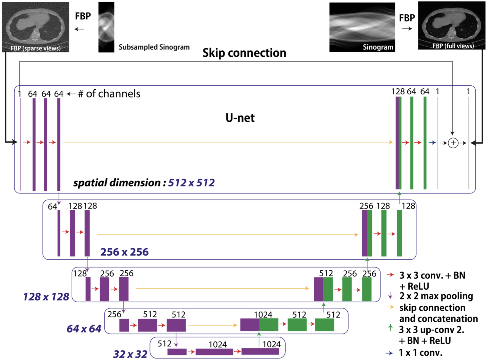
Compared to "classic" CNNs, there are **no fully-connected layers** in a U-net!

# U-Net: The Original Architecture

- Contracting path. U-Net uses normal convolutions ($3\times3$ and ReLU) and pooling to compress the image. The result is that the detailed spatial information is lost because compared to the initial size the dimension of height and width is much smaller, but also much deeper. This part of a U-Net it is similar to "classic" CNNs.
- Expansive path. This part uses transpose convolutions (or up-conv) to increase the representation size back to the size of the original input image. Transpose convolutions are a key building block of U-nets as they allow to take a small(er) input and blow it up into a larger output.
- Skip connections. These are an essential ingredient of the U-net to make the architecture work better. These are also called copy and crop.
- Output layer. This uses a $1\times1$ convolution to classify each one of the pixels in one of the classes. Taking the maximum over the number of classes provides the final classification and allows to visualise the segmentation map.

# FBPConvNet: A Post-processing Supervised Approach

K.H. Jin, M.T. McCann, E. Froustey and M. Unser
Deep Convolutional Neural Network for Inverse Problems in Imaging
IEEE Trans. Image Proc. **26**(9), 4509-4522, 2017



FBPConvNet in a nutshell:

- Initial reconstruction by FBP
- Learned denoising of the reconstructed image
- Residual U-Net architecture

# FBPConvNet: A Post-processing Supervised Approach

Aim: Define a network $\Lambda_\theta$ that processes an initial reconstruction $\widetilde{\boldsymbol{f}} = \boldsymbol{\mathcal{R}}^\dagger \boldsymbol{y}$ such that $\boldsymbol{f}^{\mathrm{gt}}$ is approximated, i.e., $\boldsymbol{f}^{\mathrm{gt}} \approx \Lambda_\theta(\widetilde{\boldsymbol{f}})$.

Some considerations:

# FBPConvNet: A Post-processing Supervised Approach

Aim: Define a network $\Lambda_\theta$ that processes an initial reconstruction $\widetilde{\boldsymbol{f}} = \boldsymbol{\mathcal{R}}^\dagger \boldsymbol{y}$ such that $\boldsymbol{f}^{\mathrm{gt}}$ is approximated, i.e., $\boldsymbol{f}^{\mathrm{gt}} \approx \Lambda_\theta(\widetilde{\boldsymbol{f}})$.

Some considerations:

- In practice, the network is trained to provide a residual $\boldsymbol{r} = \boldsymbol{f}^{\mathrm{gt}} - \widetilde{\boldsymbol{f}}$ correction to the initial reconstruction:

  $$\boldsymbol{f}^{\mathrm{gt}} \approx (\boldsymbol{I} + \Lambda_\theta)\widetilde{\boldsymbol{f}} = \widetilde{\boldsymbol{f}} + \Lambda_\theta(\widetilde{\boldsymbol{f}})$$

  That is, the network needs to be expressive enough to identify and extract noise and artifacts to be removed from the image

# FBPConvNet: A Post-processing Supervised Approach

Aim: Define a network $\Lambda_\theta$ that processes an initial reconstruction $\widetilde{\boldsymbol{f}} = \boldsymbol{\mathcal{R}}^\dagger \boldsymbol{y}$ such that $\boldsymbol{f}^{\mathrm{gt}}$ is approximated, i.e., $\boldsymbol{f}^{\mathrm{gt}} \approx \Lambda_\theta(\widetilde{\boldsymbol{f}})$.

Some considerations:

- In practice, the network is trained to provide a residual $\boldsymbol{r} = \boldsymbol{f}^{\mathrm{gt}} - \widetilde{\boldsymbol{f}}$ correction to the initial reconstruction:

  $$\boldsymbol{f}^{\mathrm{gt}} \approx (\boldsymbol{I} + \Lambda_\theta)\widetilde{\boldsymbol{f}} = \widetilde{\boldsymbol{f}} + \Lambda_\theta(\widetilde{\boldsymbol{f}})$$

  That is, the network needs to be expressive enough to identify and extract noise and artifacts to be removed from the image
- The reconstruction result depends largely on the quality of the initial reconstruction and the training data provided

# FBPConvNet: A Post-processing Supervised Approach

Aim: Define a network $\Lambda_\theta$ that processes an initial reconstruction $\widetilde{\boldsymbol{f}} = \boldsymbol{\mathcal{R}}^\dagger \boldsymbol{y}$ such that $\boldsymbol{f}^{\mathrm{gt}}$ is approximated, i.e., $\boldsymbol{f}^{\mathrm{gt}} \approx \Lambda_\theta(\widetilde{\boldsymbol{f}})$.

Some considerations:

- In practice, the network is trained to provide a residual $\boldsymbol{r} = \boldsymbol{f}^{\mathrm{gt}} - \widetilde{\boldsymbol{f}}$ correction to the initial reconstruction:

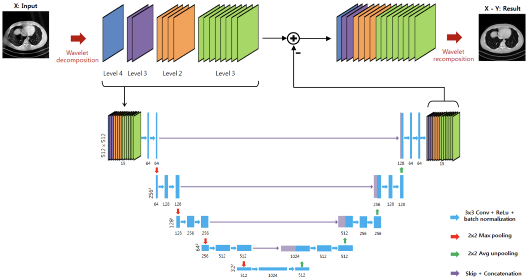
  $$\boldsymbol{f}^{\mathrm{gt}} \approx (\boldsymbol{I} + \Lambda_\theta)\widetilde{\boldsymbol{f}} = \widetilde{\boldsymbol{f}} + \Lambda_\theta(\widetilde{\boldsymbol{f}})$$

  That is, the network needs to be expressive enough to identify and extract noise and artifacts to be removed from the image
- The reconstruction result depends largely on the quality of the initial reconstruction and the training data provided
- There is no guarantee that the reconstructed images are consistent with the data, i.e., that

  $$\|\boldsymbol{\mathcal{A}}\Lambda_\theta(\widetilde{\boldsymbol{f}}) - \boldsymbol{y}\|_Y \quad \text{is small}$$

# Another Example: Multi-Scale Wavelet Domain Residual Learning

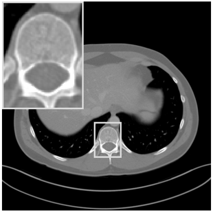
J. Gu and J.C. Ye
Multi-Scale Wavelet Domain Residual Learning for Limited-Angle CT Reconstruction, preprint, 2017



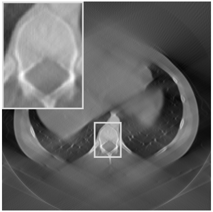
- ➤ Residual network (like FBPConvNet) but in wavelet domain
- ➤ Leveraging directional property of limited angle artifacts

- ➤ Relation to wavelet multiresolution analysis further developed in Deep Convolutional Framelets

# Some Results on Limited-Angle CT: Mayo-$60^\circ$

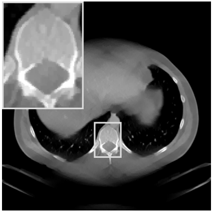


ground truth $\boldsymbol{f}^{\texttt{gt}}$



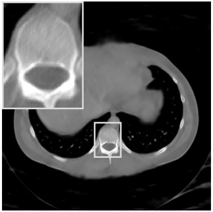
$\boldsymbol{f}_{\texttt{FBP}}$

PSNR: 17.16



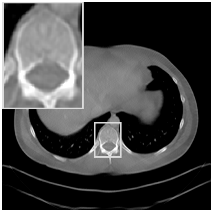
$\boldsymbol{f}_{\texttt{TV}}$

PSNR: 25.88



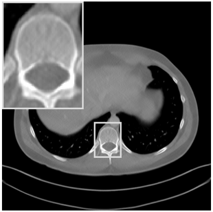
$\boldsymbol{f}_{\text{[Gu \& Ye, 2017]}}$

PSNR: 23.06



$\boldsymbol{f}_{\texttt{FBPConvNet}}$

PSNR: 27.40



$\boldsymbol{f}_{[\texttt{LtI}, \text{ 2019}]}$

PSNR: 32.77

# Learned Iterative Schemes: Unrolled Neural Networks

**Unrolling:** framework integrating a knowledge-driven model for how data are generated into a data-driven method for reconstruction via iterative schemes.

**Example:** consider linear, variational problems of the form $\operatorname{argmin}_{\boldsymbol{f}\in\mathbb{R}^n} \|\boldsymbol{\mathcal{A}}\boldsymbol{f} - \boldsymbol{y}\|_2^2$

**Algorithm** Gradient descent

```
for $k = 1, 2, 3, \ldots$ do
    $\boldsymbol{f}^{(k+1)} \leftarrow \boldsymbol{f}^{(k)} - \alpha\boldsymbol{\mathcal{A}}^\top(\boldsymbol{\mathcal{A}}\boldsymbol{f}^{(k)} - \boldsymbol{y})$
end for
```

**Algorithm** Learned gradient descent

```
for $k = 1, 2, 3, \ldots, K$ do
    $\boldsymbol{f}^{(k+1)} \leftarrow \Lambda_\theta\big(\boldsymbol{f}^{(k)}, \boldsymbol{\mathcal{A}}^\top(\boldsymbol{\mathcal{A}}\boldsymbol{f}^{(k)} - \boldsymbol{y})\big)$
    $\mathscr{R}_\theta(\boldsymbol{y}) \leftarrow \boldsymbol{f}^{(K)}$
end for
```

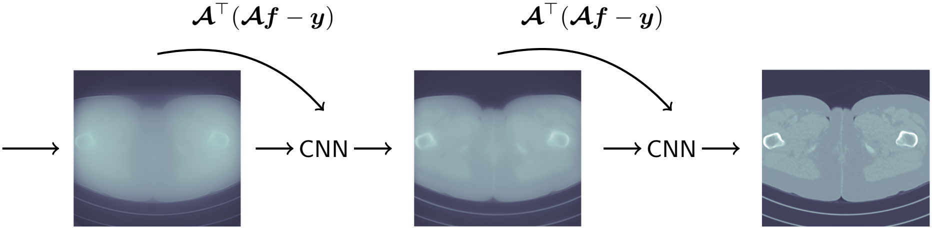
- $K = \#$ of unrolled iterates
- $\Lambda_\theta$ is a CNN
- $\theta$ learned from training data in $X \times Y$.

# Learned Gradient Descent

J. Ö. Adler and O. Öktem

Learned primal-dual reconstruction

IEEE Trans. Med. Imaging. **37**(6), 1322-1332, 2018



$\boldsymbol{\mathcal{A}}^\top(\boldsymbol{\mathcal{A}}\boldsymbol{f} - \boldsymbol{y})$ $\boldsymbol{\mathcal{A}}^\top(\boldsymbol{\mathcal{A}}\boldsymbol{f} - \boldsymbol{y})$

$\longrightarrow$ $\longrightarrow$ CNN $\longrightarrow$ $\longrightarrow$ CNN $\longrightarrow$

Unrolling a gradient descent scheme with $K = 3$ layers and linear operator $\boldsymbol{\mathcal{A}}$.

# Recall ISTA (Iterative Soft-Thresholding Algorithm)

Recall the regularized problem

$$\boldsymbol{f}^{\mathsf{WLET}} = \operatorname{argmin}\left\{\frac{1}{2}\,\|\boldsymbol{\mathcal{A}}\boldsymbol{f} - \boldsymbol{y}^{\delta}\|_2^2 + \lambda\|\boldsymbol{\mathcal{W}}\boldsymbol{f}\|_1\right\},$$

and the ISTA update:

$$\boldsymbol{f}^{(k+1)} = \boldsymbol{\mathcal{W}}^{\top} S_{\gamma} \boldsymbol{\mathcal{W}}(\boldsymbol{f}^{(k)} - \gamma\boldsymbol{\mathcal{A}}^{\top}\boldsymbol{\mathcal{A}}\boldsymbol{f}^{(k)} + \gamma\boldsymbol{\mathcal{A}}^{\top}\boldsymbol{y}^{\delta})$$

where $[S_{\beta}(\boldsymbol{x})]_i = S_{\beta}(\boldsymbol{x}_i) = \operatorname{sign}(\boldsymbol{x})(|\boldsymbol{x}| - \beta)^{+}$ is the soft-thresholding operator.

# Recall ISTA (Iterative Soft-Thresholding Algorithm)

Recall the regularized problem

$$\boldsymbol{f}^{\text{WLET}} = \operatorname{argmin}\left\{\frac{1}{2}\,\|\boldsymbol{\mathcal{A}}\boldsymbol{f} - \boldsymbol{y}^{\delta}\|_2^2 + \lambda\|\boldsymbol{\mathcal{W}}\boldsymbol{f}\|_1\right\},$$

and the ISTA update:

$$\boldsymbol{f}^{(k+1)} = \boldsymbol{\mathcal{W}}^{\top} S_{\gamma} \boldsymbol{\mathcal{W}}(\boldsymbol{f}^{(k)} - \gamma\boldsymbol{\mathcal{A}}^{\top}\boldsymbol{\mathcal{A}}\boldsymbol{f}^{(k)} + \gamma\boldsymbol{\mathcal{A}}^{\top}\boldsymbol{y}^{\delta})$$

where $[S_{\beta}(\boldsymbol{x})]_i = S_{\beta}(\boldsymbol{x}_i) = \operatorname{sign}(\boldsymbol{x})(|\boldsymbol{x}| - \beta)^{+}$ is the soft-thresholding operator.

## In general: PGD & Learned PGD

$$\boldsymbol{f}^{(k+1)} = h(\boldsymbol{f}^{(k)}, \theta) = \operatorname{prox}_{\lambda_k \varphi}\left(\boldsymbol{f}^{(k)} - \gamma_k \boldsymbol{\mathcal{A}}^{\top}(\boldsymbol{\mathcal{A}}\boldsymbol{f}^{(k)} - \boldsymbol{y}^{\delta})\right)$$

- Linear/convolutional layer: $\gamma_k \boldsymbol{\mathcal{A}}^{\top}\boldsymbol{\mathcal{A}}\boldsymbol{f}^{(k)}$ (or a neural network)
  Bias term: $\gamma_k \boldsymbol{\mathcal{A}}^{\top}\boldsymbol{y}^{\delta}$
- Activation function: $\operatorname{prox}_{\lambda_k \varphi}$ (or a neural network)

# Learned ISTA

Consider variational problems of the form

$$\underset{Z\in\mathbb{R}^m}{\operatorname{argmin}}\ \frac{1}{2}\,\|X - W_d Z\|_2^2 + \alpha\|Z\|_1$$

where $X \in \mathbb{R}^n$ is a given input vector, $W_d \in \mathbb{R}^{n\times m}$ dictionary matrix and $\alpha$ is the regularization parameter.

# Learned ISTA

Consider variational problems of the form

$$\underset{Z\in\mathbb{R}^m}{\operatorname{argmin}}\ \frac{1}{2}\,\|X - W_d Z\|_2^2 + \alpha\|Z\|_1$$

where $X \in \mathbb{R}^n$ is a given input vector, $W_d \in \mathbb{R}^{n\times m}$ dictionary matrix and $\alpha$ is the regularization parameter.

**Algorithm 1** ISTA

```
**function ISTA**$(X, Z, W_d, \alpha, L)$
    **Require:** $L >$ largest eigenvalue of $W_d^T W_d$.
    **Initialize:** $Z = 0$,
    **repeat**
        $Z = h_{(\alpha/L)}(Z - \frac{1}{L}W_d^T(W_d Z - X))$
    **until** change in $Z$ below a threshold
**end function**
```

$W_e$ is the transpose of the dictionary matrix $W_d$ and $S = W_d^\top W_d$.

**Algorithm 3** LISTA::fprop

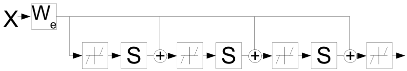
```
**LISTA :: fprop**$(X, Z, W_e, S, \theta)$
;; Arguments are passed by reference.
;; variables $Z(t)$, $C(t)$ and $B$ are saved for bprop.
$B = W_e X$; $Z(0) = h_\theta(B)$
**for** $t = 1$ to $T$ **do**
    $C(t) = B + SZ(t-1)$
    $Z(t) = h_\theta(C(t))$
**end for**
$Z = Z(T)$
```

K. Gregor and Y. LeCun

Learning fast approximations of sparse coding

27th Int. Conf. Machine Learning (ICML 2010)

# Learned ISTA



```
Algorithm 3 LISTA::fprop
  LISTA :: fprop($X, Z, W_e, S, \theta$)
  ;; Arguments are passed by reference.
  ;; variables $Z(t)$, $C(t)$ and $B$ are saved for bprop.
  $B = W_e X$; $Z(0) = h_\theta(B)$
  for $t = 1$ to $T$ do
    $C(t) = B + SZ(t-1)$
    $Z(t) = h_\theta(C(t))$
  end for
  $Z = Z(T)$
```

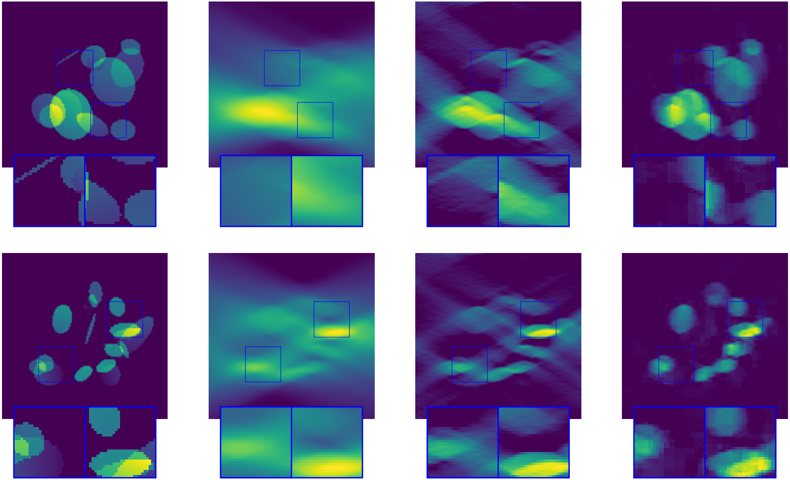
Learnable parameters $(W_e, S, \theta)$ with $\mathscr{W}_t := S = W_d^\top W_d$ weight and $b_t := W_e X$ bias so that:

$$Z(t) = h_\theta(\mathscr{W}_t Z(t-1) + b_t) \qquad \text{and} \qquad Z = (\mathscr{R}_t \circ \ldots \circ \mathscr{R}_1)(Z(0))$$

where $h_\theta$ is the shrinkage function acting as *pointwise non-linearity*.

- Time-unfolded recurrent neural network or feed-forward network in which $S$ is shared over layers
- The network is trained using training samples through back-propagation

# Some Results on Limited-Angle CT: Ellipses-$30°$



$\boldsymbol{f}^{\mathrm{gt}}$

$\boldsymbol{f}_{\mathtt{FBP}}$
PSNR: 17.83

$\boldsymbol{f}_{\mathtt{ISTA}}$
PSNR: 19.39

$\boldsymbol{f}_{\Psi\mathtt{DONet},\,2021}$
PSNR: 25.33

# Supervised VS Self-Supervised Learning

Main drawback: availability of training data, e.g., medical applications

- ✗ Need for full data with high radiation dose
- ✗ Domain shift: device differences in imaging protocols and resolution
- ✗ Global and regional data regulation policies
- ✗ Biases in the reconstruction model from differences in demographics

# Supervised VS Self-Supervised Learning

Main drawback: availability of training data, e.g., medical applications

- ✗ Need for full data with high radiation dose
- ✗ Domain shift: device differences in imaging protocols and resolution
- ✗ Global and regional data regulation policies
- ✗ Biases in the reconstruction model from differences in demographics

### Self-supervised learning

Define a model that uses **only** acquired measurement to learn a reconstruction.

- Deep Image Prior: solve $\widehat{\theta} \in \operatorname{argmin}_\theta \|\mathcal{A}\Lambda_\theta(\boldsymbol{z}) - \boldsymbol{y}\|^2$
- Noise2Noise: solve $\widehat{\theta} \in \operatorname{argmin}_\theta \mathbb{E}\left[\|\Lambda_\theta(\tilde{\boldsymbol{y}}) - \boldsymbol{y}\|^2\right]$
- Many more: Plug-and-Play, Equivariant imaging, . . .

# Learning a Regularizer: Plug and Play (PnP)

S.V. Venkatakrishnan, C.A. Bouman, and B. Wohlberg
Plug-and-play priors for model based reconstruction
IEEE GlobalSIP, 945-948, 2013

Consider the general regularized problem:

$$\underset{\boldsymbol{f}\in\mathbb{R}^n}{\operatorname{argmin}}\, D(\boldsymbol{f}) + \varphi(\boldsymbol{f}) := \left\{ \frac{\alpha}{2} \|\boldsymbol{\mathcal{A}}\boldsymbol{f} - \boldsymbol{y}^\delta\|_2^2 + \varphi(\boldsymbol{f}) \right\}.$$

Different choices of $\varphi$ promote different features in the solution. Instead of handcrafting a regularizer, we can learn its proximal operator!

# Learning a Regularizer: Plug and Play (PnP)

S.V. Venkatakrishnan, C.A. Bouman, and B. Wohlberg
Plug-and-play priors for model based reconstruction
IEEE GlobalSIP, 945-948, 2013

Consider the general regularized problem:

$$\underset{\boldsymbol{f}\in\mathbb{R}^n}{\operatorname{argmin}}\, D(\boldsymbol{f}) + \varphi(\boldsymbol{f}) := \left\{ \frac{\alpha}{2} \|\boldsymbol{\mathcal{A}}\boldsymbol{f} - \boldsymbol{y}^\delta\|_2^2 + \varphi(\boldsymbol{f}) \right\}.$$

Different choices of $\varphi$ promote different features in the solution. Instead of handcrafting a regularizer, we can learn its proximal operator!

### Plug and Play (PnP) paradigm

In, e.g., PGD replace $\operatorname{prox}_{\gamma\varphi}$ by a denoiser $\bar{\mathrm{D}}_\sigma$:

$$\boldsymbol{f}^{(k+1)} = \bar{\mathrm{D}}_\sigma(\boldsymbol{f}^{(k)} - \nabla D(\boldsymbol{f}^{(k)})),$$

where $\bar{\mathrm{D}}_\sigma$ is separately trained using pairs $(\boldsymbol{f}_i, \boldsymbol{f}_i + \boldsymbol{\epsilon}_i)$ of clear and noisy images, with $\boldsymbol{\epsilon}_i \sim \mathcal{N}(0, \sigma^2 I)$.

# Plug and Play: Proximable Denoiser

S. Hurault, A. Chambolle, A. Leclaire and N. Papadakis
Convergent plug-and-play with proximal denoiser and unconstrained regularization parameter
JMIV, 1-13, 2024

For specific choices of the denoiser, there exists an explicit expression of $\varphi$ such that $\bar{\mathrm{D}}_\sigma$ is the proximity operator of $\varphi$.

# Plug and Play: Proximable Denoiser

S. Hurault, A. Chambolle, A. Leclaire and N. Papadakis
Convergent plug-and-play with proximal denoiser and unconstrained regularization parameter
JMIV, 1-13, 2024

For specific choices of the denoiser, there exists an explicit expression of $\varphi$ such that $\bar{\mathrm{D}}_\sigma$ is the proximity operator of $\varphi$.

## Example: Gradient Step denoiser

Let $g_\sigma$ be l.s.c. and differentiable, such that

$$\bar{\mathrm{D}}_\sigma(\boldsymbol{f}) = \boldsymbol{f} - \nabla g_\sigma(\boldsymbol{f}).$$

In particular, $g_\sigma(\boldsymbol{f}) = \frac{1}{2}\left\|\boldsymbol{f} - N_\sigma(\boldsymbol{f})\right\|^2$ using a neural network $N_\sigma(\boldsymbol{f})$.

If $\nabla g_\sigma$ is $L_{g_\sigma}$-Lipschitz, $L_{g_\sigma} < 1$, there exists a $\frac{L_{g_\sigma}}{L_{g_\sigma}+1}$-weakly convex $\varphi_\sigma$, s.t.:

$$\bar{\mathrm{D}}_\sigma(\boldsymbol{f}) = \operatorname{prox}_{\varphi_\sigma}(\boldsymbol{f}), \qquad \text{where}$$

$$\phi_\sigma(\boldsymbol{f}) = \begin{cases} g_\sigma(\bar{\mathrm{D}}_\sigma^{-1}(\boldsymbol{f})) - \frac{1}{2}\|\bar{\mathrm{D}}_\sigma^{-1}(\boldsymbol{f}) - \boldsymbol{f}\|_2^2 & \text{if } \boldsymbol{f} \in \mathrm{Im}(\bar{\mathrm{D}}_\sigma) \\ +\infty & \text{otherwise.} \end{cases}$$

# Proximable Gradient Step denoiser

S. Hurault, A. Leclaire and N. Papadakis
Proximal denoiser for convergent plug-and-play optimization with nonconvex regularization, PMLR, 9483-9505, 2022

Using GS denoisers convergence has been established for PnP-PGD:

$$\boldsymbol{f}^{(k+1)} = \bar{\mathrm{D}}_\sigma(\boldsymbol{f}^{(k)} - \nabla D(\boldsymbol{f}^{(k)})).$$

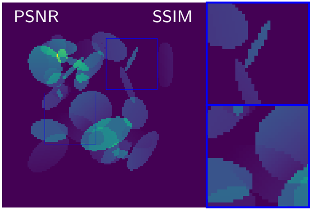
### Theorem [Hurault et al. 2022]

Assumptions:
- $D$ is bbd below, and s.t. $\nabla D$ is $L_D$-Lipschitz, with $L_D < 1$.
- $g_\sigma$ is bbd below, and s.t. $\nabla g_\sigma$ $L_{g_\sigma}$-Lipschitz, with $L_{g_\sigma} < 1$.

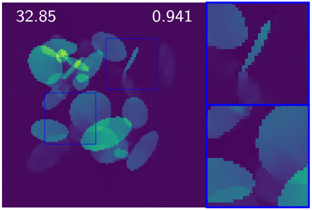
Let $F_\sigma := D + \varphi_\sigma$ with $\varphi_\sigma$ defined by GS denoiser. Then, we have:

1. $F_\sigma(\boldsymbol{f}^{(k)})$ is non-increasing and converges.
2. $\|\boldsymbol{f}^{(k+1)} - \boldsymbol{f}^{(k)}\| \to 0$ with a rate $\min_{k \leq K} \|\boldsymbol{f}^{(k+1)} - \boldsymbol{f}^{(k)}\|^2 = \mathcal{O}(1/K)$.
3. All cluster points of $(\boldsymbol{f}^{(k)})_{k \in \mathbb{N}}$ are stationary point for $F_\sigma$.
4. If $D$ and $g_\sigma$ are KL and semi-algebraic and $(\boldsymbol{f}^{(k)})_{k \in \mathbb{N}}$ is bounded, then it converges with finite length to a stationary point of $F_\sigma$.

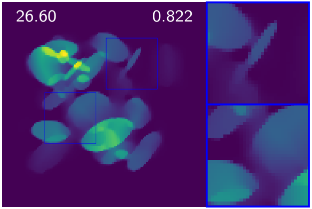
# Some Results on Limited-Angle CT: Ellipses-$60^\circ$



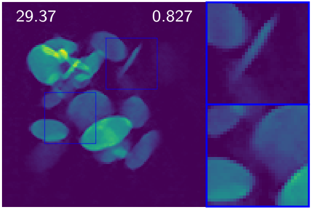
$\boldsymbol{f}^{\mathtt{gt}}$



$\boldsymbol{f}_{\mathtt{FBPConvNet}}$



$\boldsymbol{f}_{\mathtt{TV}}$



$\boldsymbol{f}_{\mathtt{PnP}}$

# Summary & Outlook

What we learned today:

- Data-driven inversion: supervised VS unsupervised
- Examples of post-processing, unrolling, PnP strategies

What I do not have time to talk about:

- Implementation details related to architecture design, choice and tuning of parameters, . . .
- There is a zoo of approaches applied to the solution of inverse problems, even if we just look at tomographic imaging!

# Some References

- Arridge, S., Maass, P., Öktem, O. and Schönlieb, C.-B., *Solving inverse problems using data-driven models*. Acta Numerica, 28, 1-174, 2019
- Goodfellow I., Bengio Y., and Courville A., *Deep Learning*, 2017
- Grohs, P. and Kutyniok, G. (Eds.), *Mathematical Aspects of Deep Learning*, 2022
- Ratti, L., *Learned reconstruction methods for inverse problems: sample error estimates*, In: Data-driven Models in Inverse Problems **31**, 2024