Towards Reliable Solutions of Inverse Problems with Deep Learning

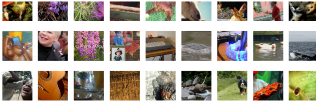
Matthias J. Ehrhardt

Department of Mathematical Sciences, University of Bath, UK

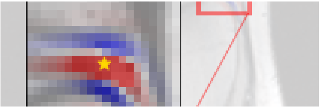
15 March 2024

Outline

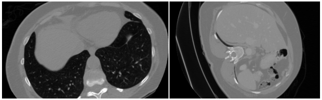
1) Machine Learning meets Inverse Problems



2) Regularization with Generative Models



3) Equivariance and Inverse Problems



Inverse problems

$$\color{green}{A}\color{red}{u} = \color{blue}{b}$$

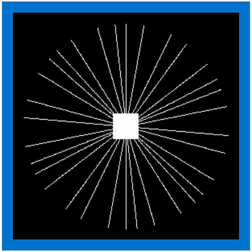
u : desired solution

b : observed data

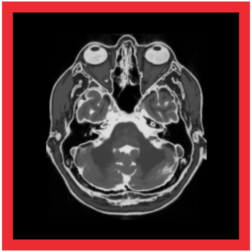
A : mathematical model

Goal: recover u given b

- MRI: Fourier transform $Au(k) = \int u(x) \exp(-ikx) dx$



$\rightarrow$



Inverse problems

$$Au = b$$

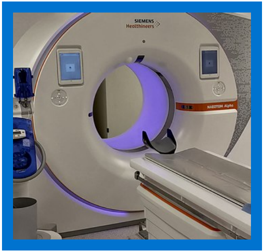
u : desired solution

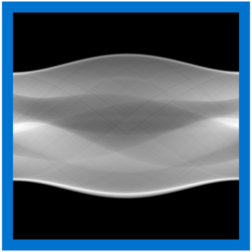
b : observed data

A : mathematical model

Goal: recover u given b

- CT: Radon / X-ray transform $Au(L) = \int_L u(x)dx$



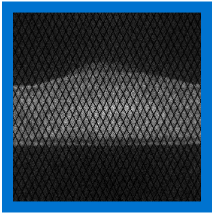


$\rightarrow$

What is the problem with Inverse Problems?

A solution may

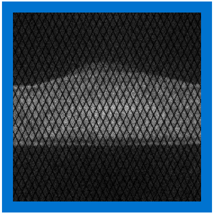
- **not exist**: define generalized solution (e.g. least squares)
- **not be unique**: select one via a-priori information (e.g. MRI)
- **be sensitive to noise**: (e.g. CT, PET)
  - Positron Emission Tomography (PET)
  - Solve $\mathbf{A}u = b$ (data: MacMillan Cancer Centre London)

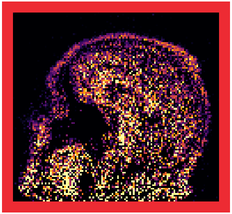


What is the problem with Inverse Problems?

A solution may

- **not exist**: define generalized solution (e.g. least squares)
- **not be unique**: select one via a-priori information (e.g. MRI)
- **be sensitive to noise**: (e.g. CT, PET)
  - Positron Emission Tomography (PET)
  - Solve $\mathbf{A}u = b$ (data: MacMillan Cancer Centre London)

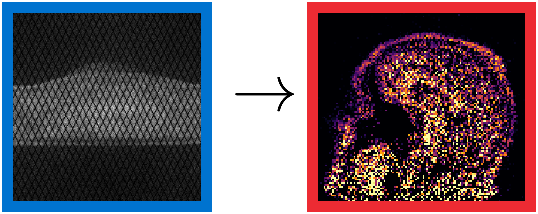




What is the problem with Inverse Problems?

A solution may

- **not exist**: define generalized solution (e.g. least squares)
- **not be unique**: select one via a-priori information (e.g. MRI)
- **be sensitive to noise**: (e.g. CT, PET)
  - Positron Emission Tomography (PET)
  - Solve $\mathbf{A}u = b$ (data: MacMillan Cancer Centre London)



- Option 1: Analytical methods
- Option 2: Variational regularization
- Option 3*: Iterative regularization
- Option 4*: Bayesian methods

Option 1: Analytical methods

$$Au = b, \quad \Phi_\lambda : b \mapsto u$$

- Find formula Φ_λ, e.g. in MRI zero-filled reconstruction, sum-of-squares, in CT or PET filtered backprojection
- Often: $\Phi_0 = A^\dagger$

Pros:

- very fast!

Cons:

- limited modelling options: forward operator
- need high-quality data: e.g. (close to) injective
- difficult to use a-priori information: e.g. nonnegativity or smoothness

Hardly used when image quality is important (except CT)

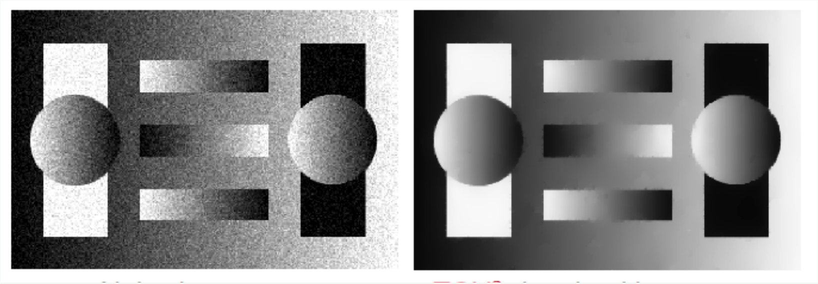
Option 2: Variational regularization

$$\Phi_\lambda(b) = \arg\min_u \{\mathcal{D}(Au, b) + \lambda \mathcal{R}(u)\}$$

$\mathcal{D}$ measures **fidelity** between Au and b, related to noise statistics

$\mathcal{R}$ **regularizer** penalizes unwanted features and ensures stability;
e.g. TV Rudin, Osher, Fatimi '92 $\mathcal{R}(u) = \|\nabla u\|_1$,
TGV Bredies, Kunisch, Pock '10 $\mathcal{R}(u) = \inf_v \|\nabla u - v\|_1 + \beta \|\nabla v\|_1$

$\lambda \geq 0$ **regularization parameter** balances fidelity and regularization



Option 2: Variational regularization (cont)

$$\Phi_\lambda(b) = \arg\min_u \{\mathcal{D}(Au, b) + \lambda\mathcal{R}(u)\}$$

- Only theoretical. Need to find algorithm (u^k) such that
$$\Phi_\lambda(b) := \lim_{k\to\infty} u^k$$

- Proximal Gradient Descent / Forward-Backward Splitting
Bauschke and Combettes '11, Beck '17 ...
$$u^{k+1} = \mathrm{prox}_{\tau_k\lambda\mathcal{R}}(u^k - \tau_k \nabla\mathcal{E}(u^k))$$
$\mathcal{E}(u) = \mathcal{D}(Au, b)$

proximal operator Moreau '62

$$\mathrm{prox}_f(z) := \arg\min_u \left\{\frac{1}{2}\|u - z\|^2 + f(u)\right\}$$

Iterate: fit data, denoise

Option 2: Variational regularization (cont)

$$\Phi_\lambda(b) = \arg\min_u \{\mathcal{D}(Au, b) + \lambda\mathcal{R}(u)\}$$

Pros:

- good modelling: forward operator, data fit and regularizer provide a lot of freedom
- data quality can be poor if exploiting a-priori knowledge
- a lof of theory available

Cons:

- difficult to choose regularisation parameter λ
- slow: many evaluations of A and A^* **ongoing research**
- modelling simple: TV, TGV work great on geometric phantoms, room for improvement for real data

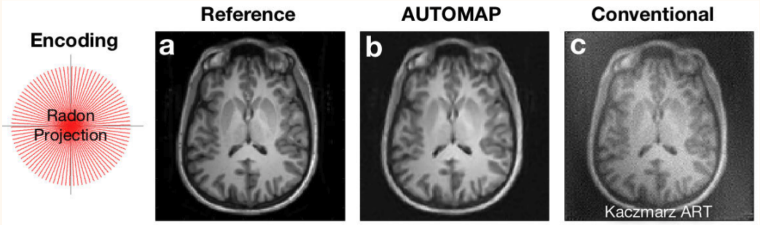
difficult to include more data: what does a **typical** reconstruction look like?

Machine Learning meets Inverse Problems (i.e. mostly deep learning)

“Analytic methods” meet Deep Learning

- automap Zhu et al. '18, Nature paper with 1600+ citations
  - ignore physical modelling (i.e. A)

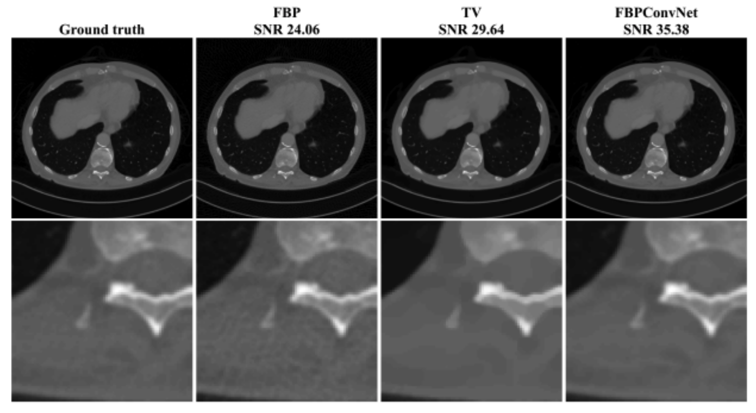
$$\Phi(b) = \mathcal{N}_\theta(b)$$



“Analytic methods” meet Deep Learning

- automap Zhu et al. ’18, Nature paper with 1600+ citations
  - ignore physical modelling (i.e. A) $\Phi(b) = \mathcal{N}_\theta(b)$
- learned postprocessing, e.g. Jin et al. ’17, 2000+ citations
  - rough recon with physical model, then apply neural network

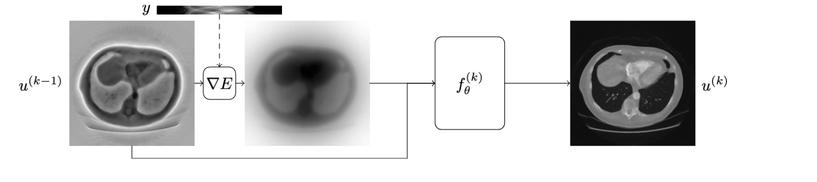
$$\Phi(b) = \mathcal{N}_\theta(A^\dagger b)$$



"Analytic methods" meet Deep Learning

- automap Zhu et al. '18, Nature paper with 1600+ citations
  - ignore physical modelling (i.e. A) $\Phi(b) = \mathcal{N}_\theta(b)$
- learned postprocessing, e.g. Jin et al. '17, 2000+ citations
  - rough recon with physical model, then apply neural network $\Phi(b) = \mathcal{N}_\theta(A^\dagger b)$
- unrolling, e.g. Gregor and Le Cun '10, Adler and Öktem '17
  - take few iterations of algorithm and replace prox with neural network

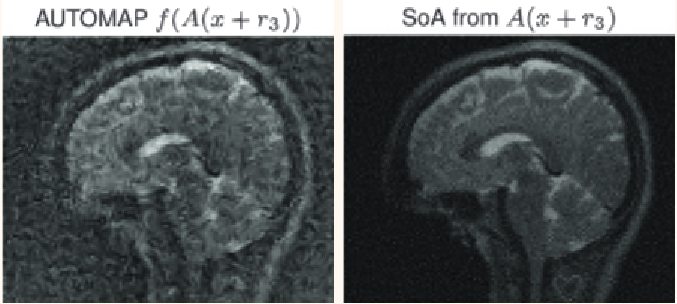
$$\Phi(b) = u^K,\ u^{k+1} = \mathcal{N}_\theta^k(u^k - \tau_k \nabla \mathcal{E}(u^k))$$



“Analytic methods” meet Deep Learning

- automap Zhu et al. '18, Nature paper with 1600+ citations
  - ignore physical modelling (i.e. A) $\Phi(b) = \mathcal{N}_\theta(b)$
- learned postprocessing, e.g. Jin et al. '17, 2000+ citations
  - rough recon with physical model, then apply neural network $\Phi(b) = \mathcal{N}_\theta(A^\dagger b)$
- unrolling, e.g. Gregor and Le Cun '10, Adler and Öktem '17
  - take few iterations of algorithm and replace prox with neural network $\Phi(b) = u^K, u^{k+1} = \mathcal{N}_\theta^k(u^k - \tau_k \nabla \mathcal{E}(u^k))$

Not as stable as pre-deep learning approaches Antun et al. '19



Variational regularization meets Deep Learning

Idea: learn a regularizer R_θ for variational regularization

- Exploit **pretrained** network, e.g. **denoiser** Romano et al. '17

$$R(u) = \frac{1}{2} u^T (u - \mathcal{N}_\theta(u))$$

Variational regularization meets Deep Learning

Idea: learn a regularizer R_θ for variational regularization

- Exploit **pretrained** network, e.g. **denoiser** Romano et al. '17

$$R(u) = \frac{1}{2} u^T (u - \mathcal{N}_\theta(u))$$

- Train **directly before reconstruction**, e.g.
  - if "good" images (u_k) and and "bad" images (v_k) are available Benning et al. '17, choose parameters θ to minimize

$$\mathbb{E}_u R_\theta(u) - \mathbb{E}_v R_\theta(v)$$

    Connected to Wasserstein distance between (u_k) and (v_k) if R_θ is 1-Lipschitz Lunz et al. '19, e.g. $v = A^\dagger b$.
  - input-convex neural networks Mukherjee et al. '20

Variational regularization meets Deep Learning

Idea: learn a regularizer R_θ for variational regularization

- Exploit **pretrained** network, e.g. **denoiser** Romano et al. '17

$$R(u) = \frac{1}{2} u^T (u - \mathcal{N}_\theta(u))$$

- Train **directly before reconstruction**, e.g.
  - if "good" images (u_k) and and "bad" images (v_k) are available Benning et al. '17, choose parameters θ to minimize

$$\mathbb{E}_u R_\theta(u) - \mathbb{E}_v R_\theta(v)$$

    Connected to Wasserstein distance between (u_k) and (v_k) if R_θ is 1-Lipschitz Lunz et al. '19, e.g. $v = A^\dagger b$.
  - input-convex neural networks Mukherjee et al. '20
- **Train for reconstruction**, e.g. **bilevel learning**:

$$\min_\theta \mathbb{E}_{u^*,b} \|\Phi_\theta(b) - u^*\|^2 \quad \Phi_\theta(b) = \arg\min_u \{D(Au, b) + R_\theta(u)\}$$

Summary

What to learn? I.e. network architecture

- deep learning and inverse problems can be combined in **various ways**
- directly using the network ("analytic" methods) can be **unstable**
- incorporating **more structure** (e.g. variational regularization) or information (e.g. A) makes the approach **more stable and needs less data**

What to learn from? I.e. training data

- Supervised: end-to-end, bilevel learning (u_i^*, b_i), potentially using A
- Unsupervised: (u_i^*), negative examples (v_i)
- Semi-Supervised: (u_i^*), (b_i), potentially using A

Regularization with Generative Models

Generative Regularizers

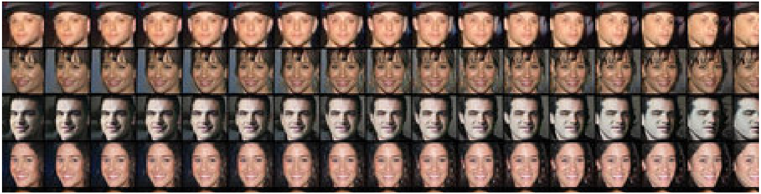


Image by Hu et al. '20

- Given a generative model $G_\theta : Z \to U$ (e.g. AE, VAE, GAN), one can define a **generative regularizer** Duff et al. JMIV '23, e.g.

$$R(u) = \inf_z \left\{ \frac{1}{2} \| u - G_\theta(z) \|_2^2 + S(z) \right\}$$

- A variant with **hard constraints** has been used in Bora et al. '17

$$R(u) = \inf_z \iota_{\{0\}}(u - G_\theta(z))$$

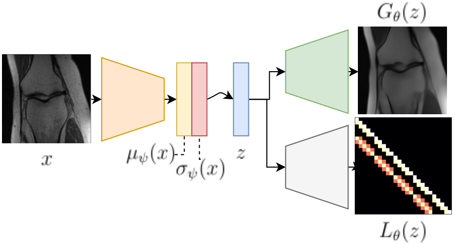
- In both cases: **only the mean** is modelled

Modelling the Covariance Duff et al. PMB '23

- Motivated by Dorta et al. '18, we use the regularizer

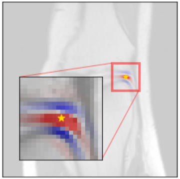
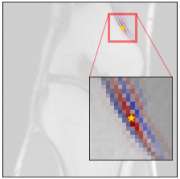
$$R(u) = \inf_z \left\{ \log\det(\Sigma(z)) + \frac{1}{2}\|u - G(z)\|^2_{\Sigma^{-1}(z)} + \frac{1}{2}\|z\|_2^2 \right\}$$

This is related to $u \propto \mathcal{N}(G(z), \Sigma(z))$ and $z \propto \mathcal{N}(0, I)$.



Margaret Duff

- Visualization of learned positive and negative covariance.

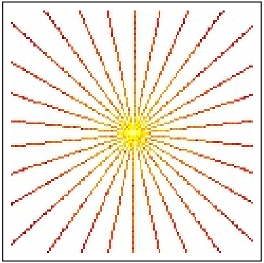
Example: Magnetic Resonance Imaging (MRI)

MRI Reconstruction

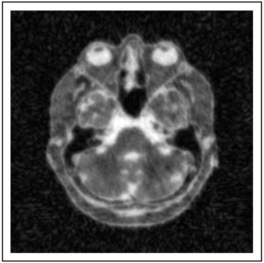
Fourier transform F, sampling $Sw = (w_i)_{i\in\Omega}$

$$\min_{u}\left\{\sum_{i\in\Omega}|(Fu)_i - b_i|^2\right\}$$

MRI scanner



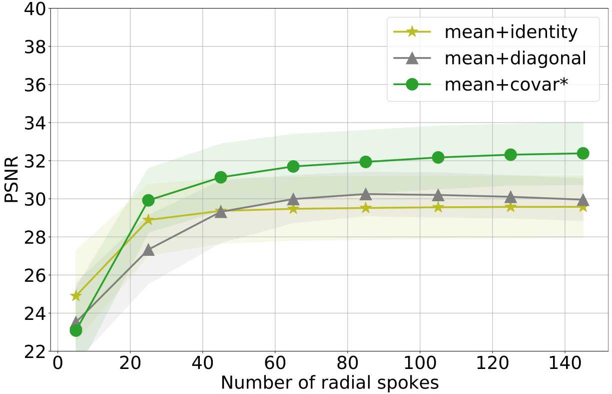
sampling S^*y



minimizer

Comparison: Covariance Models

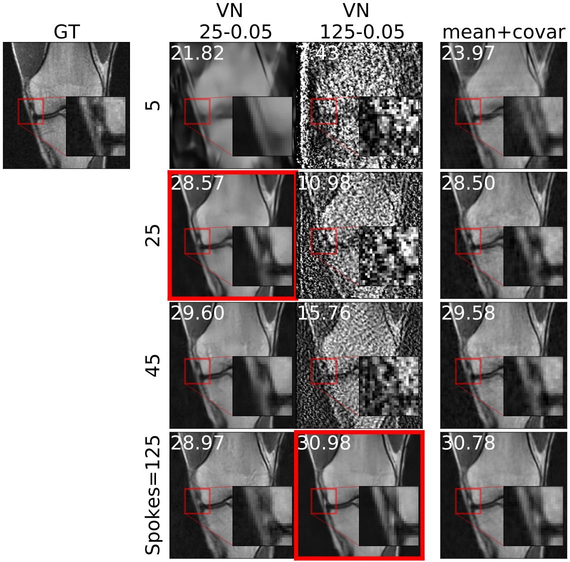
- constant diagonal (identity)
- varying diagonal (diagonal)
- proposed (covar)



- In any case, the proposed model appears superior.

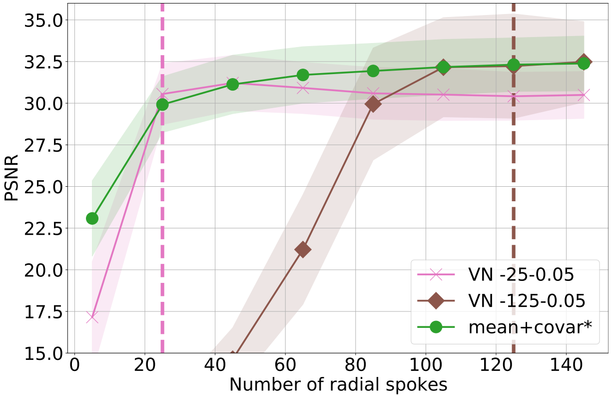
Comparison: End-to-end Learning

- Compare to Variational Network (VN) Hammernik et al. '18 trained for specific sampling and noise (indicated in red).



Comparison: End-to-end Learning (cont)

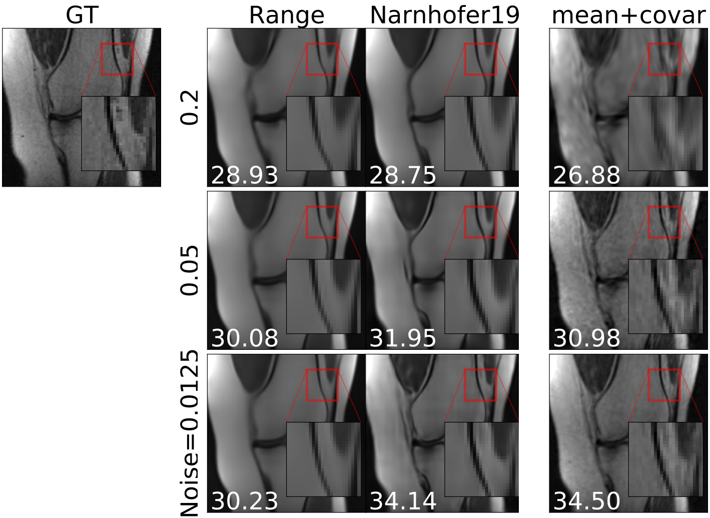
Compare to Variational Network (VN) Hammernik et al. '18 trained for specific sampling and noise (dashed lines).



- Similar peak performance but proposed model **generalizes better** to unseen settings.

Comparison: Other unsupervised methods

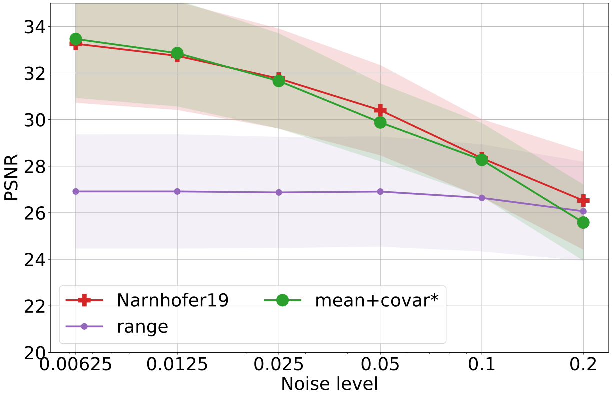
- Compare to Bora et al. '17 (Range) which restricts to the range.
- Compare to Narnhofer et al. '19 which uses an Inverse GAN.



- Bora et al. '17, Narnhofer et al. '19 produce **smoother solutions**.

Comparison: Other unsupervised methods (cont)

- Compare to Bora et al. '17 (Range) which restricts to the range.
- Compare to Narnhofer et al. '19 which uses an Inverse GAN.



- Better than Bora et al. '17. Similar to Narnhofer et al. '19.

Equivariance and Inverse Problems

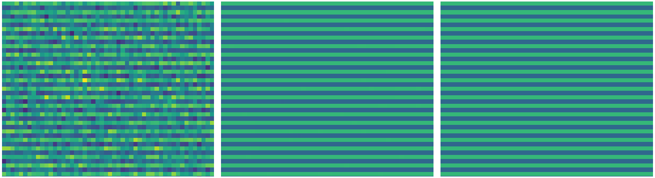
What happens when data is rotated?

Example: R rotation, Φ denoising network

$$\Phi(Rb) \stackrel{?}{=} R\Phi(b)$$

Ferdia Sherry

Training data



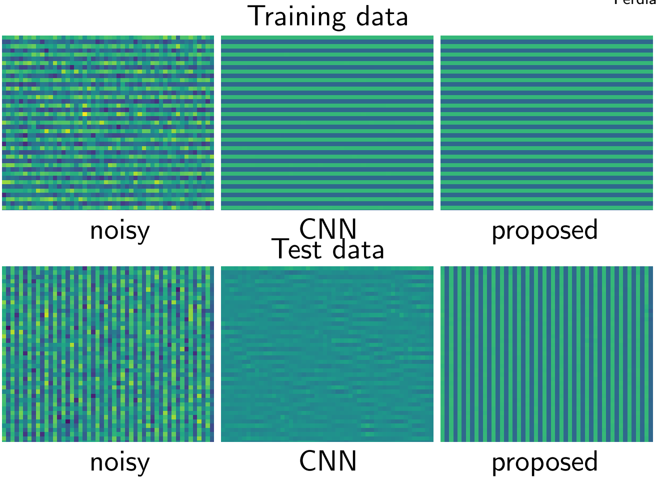
noisy CNN proposed

What happens when data is rotated?

Example: R rotation, Φ denoising network

$$\Phi(Rb) \stackrel{?}{=} R\Phi(b)$$

Ferdia Sherry



How to get “equivariant” mappings?

$$\Phi(Rb) = R\Phi(b)$$

- **equivariance by learning**: e.g. data augmentation $(b_i, u_i)_i$ becomes $(R_i b_i, R_i u_i)_i$
  - ✓ **simple to implement** for image-based tasks (e.g. denoising, image segmentation etc)
  - ✗ potentially **computationally costly**: larger training data
  - ✗ **no guarantees** to generalize to test data
  - ✗ **not always easy/possible** (for inverse problems only viable in simulations or if data is not paired)

How to get "equivariant" mappings?

$$\Phi(Rb) = R\Phi(b)$$

- **equivariance by learning**: e.g. data augmentation $(b_i, u_i)_i$ becomes $(R_i b_i, R_i u_i)_i$
  - ✓ **simple to implement** for image-based tasks (e.g. denoising, image segmentation etc)
  - ✗ potentially **computationally costly**: larger training data
  - ✗ **no guarantees** to generalize to test data
  - ✗ **not always easy/possible** (for inverse problems only viable in simulations or if data is not paired)
- **equivariance by design**
  - ✓ **mathematical guarantees**
  - ✗ **not trivial** to do

Provable equivariant neural networks have been studied a lot for segmentation, classification, denoising etc

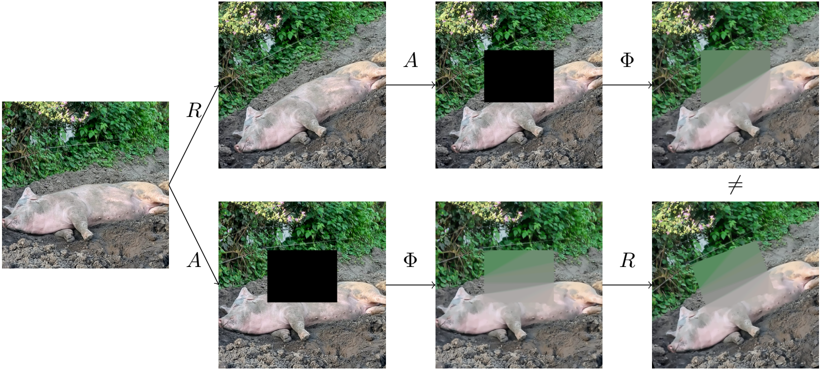
Bekkers et al. '18, Weiler and Cesa '19, Cohen and Welling '16, Dieleman et al. '16, Sosnovik et al. '19, Worall and Welling '19, ...

Equivariance and inverse problems

- inverse problem $Au = b$, solution operator: $\Phi : Y \to X$
- **Hope** $\Phi \circ A$ is equivariant, e.g. $R \circ \Phi \circ A = \Phi \circ A \circ R$

Equivariance and inverse problems

- inverse problem $Au = b$, solution operator: $\Phi : Y \to X$
- **Hope** $\Phi \circ A$ is equivariant, e.g. $R \circ \Phi \circ A = \Phi \circ A \circ R$
- $\Phi \circ A$ generally **not equivariant**. TV inpainting



Group acting on images

- Example groups (image from Chen et al. '23):

- $\overline{G} = \mathbb{R}^n \rtimes H$, $\quad H$ subgroup of the general linear group $\mathrm{GL}(n)$
- $g \cdot x = Rx + t, g = (t, R) \in \overline{G}, t \in \mathbb{R}^n, R \in H$
- $(g \cdot u)(x) = u(R^{-1}(x - t))$

This includes Weiler and Cesa '19

- **Translations:** $H = \{e\}$
- **Roto-Translations:** $H = \mathrm{SO}(n)$
- **Finite Roto-Translations** $H = Z_M$ (finite subgroup of $\mathrm{SO}(n)$)

Invariant functional implies equivariant prox

Theorem Celledoni et al. '21
Let $X = L^2(\Omega)$, J **invariant**: $J(g \cdot u) = J(u)$. Then prox_J is **equivariant**, i.e. for all $u \in X$

$$\text{prox}_J(g \cdot u) = g \cdot \text{prox}_J(u).$$

- Total variation (and higher order variants) is invariant to rigid motion
- Natural condition on networks for unrolled algorithms

How to construct equivariant networks?

Proposition Let G be any group and Φ and Ψ equivariant.

- The **composition** $\Phi \circ \Psi$ is equivariant.
- The **sum** $\Phi + \Psi$ is equivariant.
- The **identity** $u \mapsto u$ is equivariant.

Next slide There are non-trivial $\overline{G}$-equivariant linear operators.

Proposition Let G be any group and $(\Phi u)(x) = u(x) + b(x)$. Φ is equivariant if b is invariant, i.e. $g \cdot b = b$.

Proposition There are $\overline{G}$-equivariant nonlinearities.

Construct $\overline{G}$-equivariant neural networks the usual way:

- layers $\Phi = \Phi_n \circ \cdots \circ \Phi_1$
- $\Phi(u) = \sigma(Au + b)$
- ResNet $\Phi(u) = u + \sigma(Au + b)$

Equivariant linear functions ($\pi_X \equiv id$)

In a nutshell: Linear $\overline{G}$-equivariant operators are convolutions with a kernel satisfying an additional constraint.

Theorem paraphrasing e.g. Weiler and Cesa '19

Let X, Y be function spaces, e.g. $X = L^2(\mathbb{R}^n, \mathbb{R}^m)$, $Y = L^2(\mathbb{R}^n, \mathbb{R}^M)$. The linear operator $\Phi : X \to Y$,

$$\Phi f(x) = \int K(x, y) f(y) dy$$

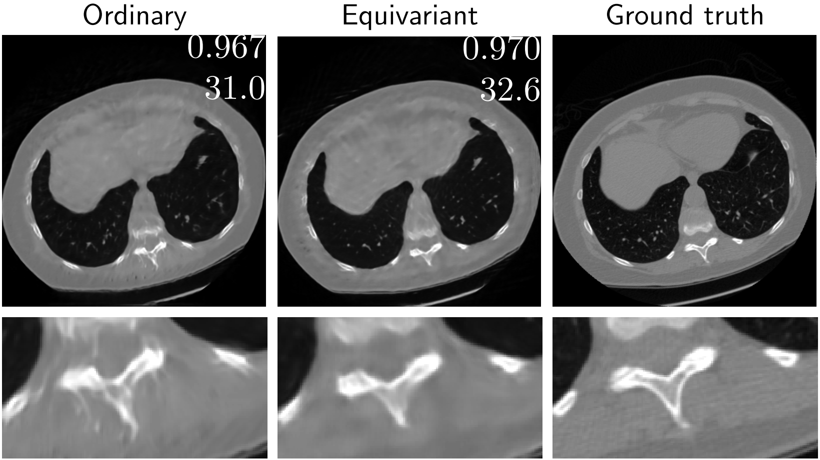
with $K : \mathbb{R}^n \to \mathbb{R}^{M \times m}$ is $\overline{G}$-**equivariant** iff there is a k such that

$$\Phi f(x) = \int k(x - y) f(y) dy$$

and k is H-invariant, i.e. for all $R \in H$, $x \in \mathbb{R}^n$: $k(Rx) = k(x)$.

CT Results

- LIDC-IDRI data set, 5000+200+1000 images, 50 views
- Equivariant = roto-translations; Ordinary = translations



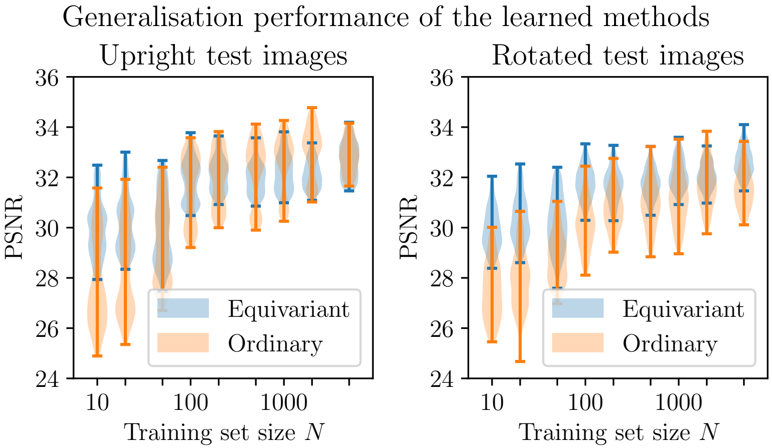
- **higher** SSIM and PSNR
- **fewer** artefacts and **finer** details

CT Results Celledoni et al., Inverse Problems, '21.

Equivariant = roto-translations; Ordinary = translations

Equivariant improves upon Ordinary:

- **small** training sets
- **unseen** orientations



Conclusions

- **Generative regularizers**: modelling of prior correlations
  - Unsupervised model: **no paired data** required
  - Learning independent of inverse problem: **generalization**
- Exploiting **equivariance**
  - natural condition when **proximal operators** are replaced
  - needs **less data**
  - **no extra computational cost** at test time