# The Mathematics of Deep Learning

Gitta Kutyniok

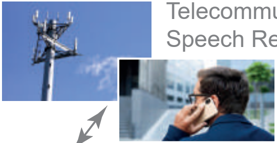
(Ludwig-Maximilians-Universität München)

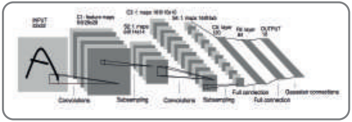
8th European Congress of Mathematics
Portorož, Slovenia, June 20 – 26, 2021

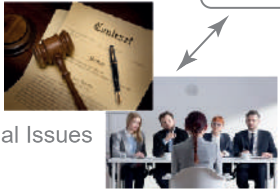
# The Dawn of Deep Learning in Public Life

Self-Driving Cars

Telecommunication/
Speech Recognition







Legal Issues

Health Care

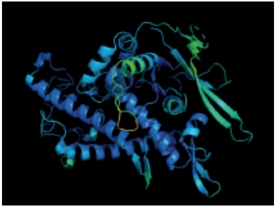
# Spectacular Success in Science

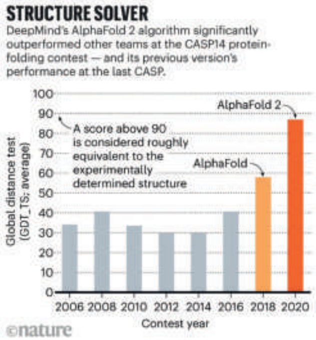
NEWS · 30 NOVEMBER 2020

## 'It will change everything': DeepMind's AI makes gigantic leap in solving protein structures

Google's deep-learning program for determining the 3D shapes of proteins stands to transform biology, say scientists.

*Nature* **588**, 203-204 (2020)





# Impact on Mathematical Problem Settings

**Some Examples:**

- Inverse Probleme/Imaging Science *(2012–)*
  - $\rightsquigarrow$ *Denoising*
  - $\rightsquigarrow$ *Edge Detection*
  - $\rightsquigarrow$ *Inpainting*
  - $\rightsquigarrow$ *Classification*
  - $\rightsquigarrow$ *Superresolution*
  - $\rightsquigarrow$ *Limited-Angle Computed Tomography*
  - $\rightsquigarrow$ ...

# Impact on Mathematical Problem Settings

## Some Examples:

- Inverse Probleme/Imaging Science *(2012–)*
  $\rightsquigarrow$ *Denoising*
  $\rightsquigarrow$ *Edge Detection*
  $\rightsquigarrow$ *Inpainting*
  $\rightsquigarrow$ *Classification*
  $\rightsquigarrow$ *Superresolution*
  $\rightsquigarrow$ *Limited-Angle Computed Tomography*
  $\rightsquigarrow$ ...

- Numerical Analysis of Partial Differential Equations *(2017–)*
  $\rightsquigarrow$ *Black-Scholes PDE*
  $\rightsquigarrow$ *Allen-Cahn PDE*
  $\rightsquigarrow$ *Parametric PDEs*
  $\rightsquigarrow$ ...

# Deep Learning = Alchemy?

AAAS | Science

## AI researchers allege that machine learning is alchemy

By Matthew Hutson | May. 3, 2018 , 11:15 AM

Ali Rahimi, a researcher in artificial intelligence (AI) at Google in San Francisco, California, took a swipe at his field last December—and received a 40-second ovation for it. Speaking at an AI conference, Rahimi charged that machine learning algorithms, in which computers learn through trial and error, have become a form of "alchemy." Researchers, he said, do not know why some algorithms work and others don't, nor do they have rigorous criteria for choosing one AI architecture over another. Now, in a paper presented on 30 April at the International Conference on Learning Representations in Vancouver, Canada, Rahimi and his collaborators document examples of what they see as the alchemy problem and offer prescriptions for bolstering AI's rigor.

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

# Problem with Trustworthiness

By Linda Geddes 5th December 2018

Computers can be made to see a sea turtle as a gun or hear a concerto as someone's voice, which is raising concerns about using artificial intelligence in the real world.

MACHINE MINDS | ARTIFICIAL INTELLIGENCE BBC

# Missing Mathematical Foundation

SIAM NEWS **MAY 2017**

Research | May 01, 2017

## Deep, Deep Trouble

### Deep Learning's Impact on Image Processing, Mathematics, and Humanity

By Michael Elad

I am really confused. I keep changing my opinion on a daily basis, and I cannot seem to settle on one solid view of this puzzle. No, I am not talking about world politics or the current U.S. president, but rather something far more critical to humankind, and more specifically to our existence and work as engineers and researchers. I am talking about...**deep learning**.

# Role of Mathematics

**Two Key Challenges for Mathematics:**

*Mathematics for Deep Learning!*

- Can we derive a deep mathematical understanding of deep learning?
- How can we make deep learning more robust?
- ...

*Deep Learning for Mathematics!*

- How can we use deep learning to improve imaging science?
- Can we develop superior PDE solvers via deep learning?
- ...

*Delving Deeper into Deep Neural Networks...*

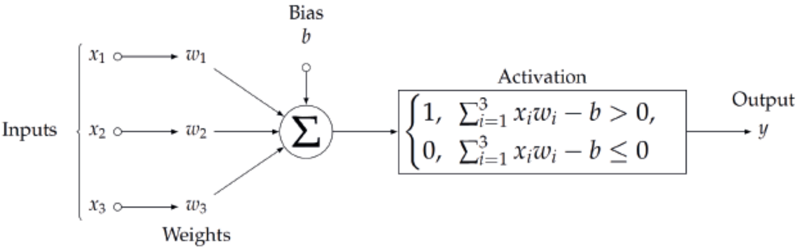
# First Appearance of Neural Networks

**Key Task of McCulloch and Pitts (1943):**

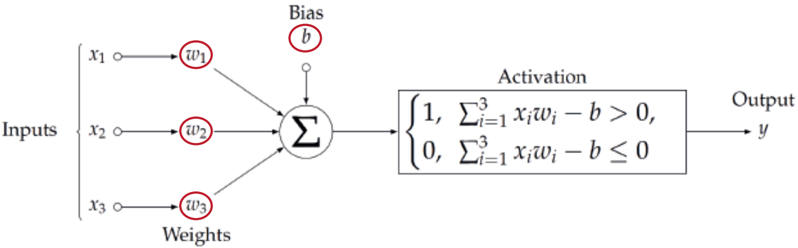
- Develop an algorithmic approach to learning.
- Mimicking the functionality of the human brain.

*Goal: Artifical Intelligence!*

# Artificial Neurons




# Artificial Neurons




# Artificial Neurons

**Definition:** An *artificial neuron* with *weights* $w_1, ..., w_n \in \mathbb{R}$, *bias* $b \in \mathbb{R}$ and *activation function* $\rho : \mathbb{R} \to \mathbb{R}$ is defined as the function $f : \mathbb{R}^n \to \mathbb{R}$ given by

$$f(x_1, ..., x_n) = \rho\left(\sum_{i=1}^{n} x_i w_i - b\right) = \rho(\langle x, w \rangle - b),$$

where $w = (w_1, ..., w_n)$ and $x = (x_1, ..., x_n)$.

# Artificial Neurons

**Definition:** An *artificial neuron* with *weights* $w_1, \dots, w_n \in \mathbb{R}$, *bias* $b \in \mathbb{R}$ and *activation function* $\rho : \mathbb{R} \to \mathbb{R}$ is defined as the function $f : \mathbb{R}^n \to \mathbb{R}$ given by

$$f(x_1, \dots, x_n) = \rho\left(\sum_{i=1}^{n} x_i w_i - b\right) = \rho(\langle x, w\rangle - b),$$

where $w = (w_1, \dots, w_n)$ and $x = (x_1, \dots, x_n)$.

**Examples of Activation Functions:**

- Heaviside function $\rho(x) = \begin{cases} 1, & x > 0, \\ 0, & x \leq 0. \end{cases}$
- Sigmoid function $\rho(x) = \frac{1}{1+e^{-x}}$.
- *Rectifiable Linear Unit (ReLU)* $\rho(x) = \max\{0, x\}$.

# Affine Linear Maps and Weights

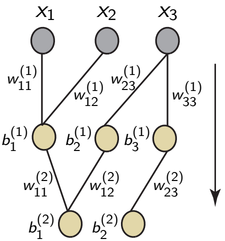
**Remark:** Concatenating artificial neurons leads to *compositions of affine linear maps and activation functions*.

**Example:** The following part of a neural network is given by

$$\Phi : \mathbb{R}^3 \to \mathbb{R}^2, \quad \Phi(x) = W^{(2)}\rho(W^{(1)}x + b^{(1)}) + b^{(2)}.$$

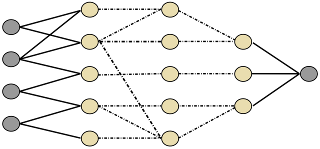
$$W^{(1)} = \begin{pmatrix} w_{11}^{(1)} & w_{12}^{(1)} & 0 \\ 0 & 0 & w_{23}^{(1)} \\ 0 & 0 & w_{33}^{(1)} \end{pmatrix}$$

$$W^{(2)} = \begin{pmatrix} w_{11}^{(2)} & w_{12}^{(2)} & 0 \\ 0 & 0 & w_{23}^{(2)} \end{pmatrix}$$




$\rightsquigarrow$ Sparse matrices lead to *sparse connectivity*!

# Definition of a Deep Neural Network



**Definition:**
Assume the following notions:

- $d \in \mathbb{N}$: Dimension of input layer.
- $L$: Number of layers.
- $\rho : \mathbb{R} \to \mathbb{R}$: (Non-linear) function called *activation function*.
- $T_\ell : \mathbb{R}^{N_{\ell-1}} \to \mathbb{R}^{N_\ell}$, $\ell = 1, \dots, L$, where $T_\ell x = W^{(\ell)} x + b^{(\ell)}$

Then $\Phi : \mathbb{R}^d \to \mathbb{R}^{N_L}$ given by

$$\Phi(x) = T_L \rho(T_{L-1} \rho(\dots \rho(T_1(x))), \quad x \in \mathbb{R}^d,$$

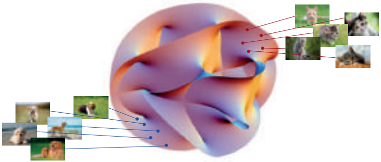
is called *(deep) neural network (DNN)*.

# Training of Deep Neural Networks

**High-Level Set Up:**

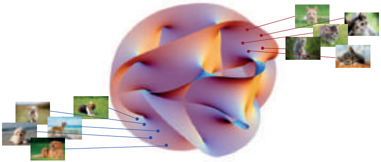
- Samples $(x_i, f(x_i))_{i=1}^m$ of a function such as $f : \mathcal{M} \to \{1, 2, \ldots, K\}$.

  $\rightsquigarrow$ *Training- and test data set.*

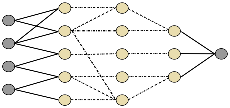


# Training of Deep Neural Networks

**High-Level Set Up:**



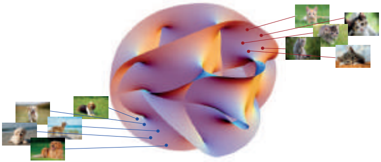
- Samples $(x_i, f(x_i))_{i=1}^m$ of a function such as $f : \mathcal{M} \to \{1, 2, \ldots, K\}$.
  $\rightsquigarrow$ *Training- and test data set.*



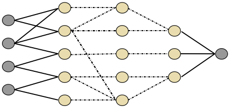
- Select an architecture of a deep neural network, i.e., a choice of $d$, $L$, $(N_\ell)_{\ell=1}^L$, and $\rho$.
  *Sometimes selected entries of the matrices $(W^{(\ell)})_{\ell=1}^L$, i.e., weights, are set to zero at this point.*

# Training of Deep Neural Networks

**High-Level Set Up:**



- Samples $(x_i, f(x_i))_{i=1}^m$ of a function such as $f : \mathcal{M} \to \{1, 2, \ldots, K\}$.
  $\rightsquigarrow$ *Training- and test data set.*



- Select an architecture of a deep neural network, i.e., a choice of $d$, $L$, $(N_\ell)_{\ell=1}^L$, and $\rho$.
  *Sometimes selected entries of the matrices $(W^{(\ell)})_{\ell=1}^L$, i.e., weights, are set to zero at this point.*
- Learn the affine-linear functions $(T_\ell)_{\ell=1}^L = (W^{(\ell)} \cdot + b^{(\ell)})_{\ell=1}^L$ by

$$\min_{(W^{(\ell)}, b^{(\ell)})_\ell} \sum_{i=1}^m \mathcal{L}(\Phi_{(W^{(\ell)}, b^{(\ell)})_\ell}(x_i), f(x_i)) + \lambda \mathcal{R}((W^{(\ell)}, b^{(\ell)})_\ell)$$

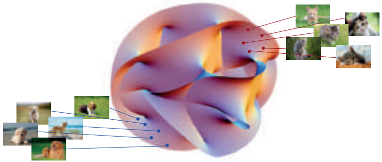
  yielding the network $\Phi_{(W^{(\ell)}, b^{(\ell)})_\ell} : \mathbb{R}^d \to \mathbb{R}^{N_L}$,

$$\Phi_{(W^{(\ell)}, b^{(\ell)})_\ell}(x) = T_L \rho(T_{L-1} \rho(\ldots \rho(T_1(x)))).$$

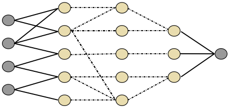
  *This is often done by stochastic gradient descent.*

# Training of Deep Neural Networks

## High-Level Set Up:



- Samples $(x_i, f(x_i))_{i=1}^m$ of a function such as $f : \mathcal{M} \to \{1, 2, \ldots, K\}$.
  $\rightsquigarrow$ *Training- and test data set.*



- Select an architecture of a deep neural network, i.e., a choice of $d$, $L$, $(N_\ell)_{\ell=1}^L$, and $\rho$.
  *Sometimes selected entries of the matrices $(W^{(\ell)})_{\ell=1}^L$, i.e., weights, are set to zero at this point.*
- Learn the affine-linear functions $(T_\ell)_{\ell=1}^L = (W^{(\ell)} \cdot + b^{(\ell)})_{\ell=1}^L$ by

$$\min_{(W^{(\ell)}, b^{(\ell)})_\ell} \sum_{i=1}^m \mathcal{L}(\Phi_{(W^{(\ell)}, b^{(\ell)})_\ell}(x_i), f(x_i)) + \lambda \mathcal{R}((W^{(\ell)}, b^{(\ell)})_\ell)$$

yielding the network $\Phi_{(W^{(\ell)}, b^{(\ell)})_\ell} : \mathbb{R}^d \to \mathbb{R}^{N_L}$,

$$\Phi_{(W^{(\ell)}, b^{(\ell)})_\ell}(x) = T_L \rho(T_{L-1} \rho(\ldots \rho(T_1(x)))).$$

*This is often done by stochastic gradient descent.*

*Goal:* $\Phi_{(W^{(\ell)}, b^{(\ell)})_\ell} \approx f$

# Second Appearance of Neural Networks

**Key Observations by Y. LeCun et al. (around 2000):**

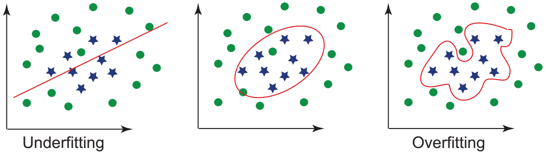
- Drastic improvement of computing power.
  $\rightsquigarrow$ *Networks with hundreds of layers can be trained.*
  $\rightsquigarrow$ *Deep Neural Networks!*
- Age of Data starts.
  $\rightsquigarrow$ *Vast amounts of training data is available.*

# Second Appearance of Neural Networks

## Key Observations by Y. LeCun et al. (around 2000):

- Drastic improvement of computing power.
  $\rightsquigarrow$ *Networks with hundreds of layers can be trained.*
  $\rightsquigarrow$ *Deep Neural Networks!*
- Age of Data starts.
  $\rightsquigarrow$ *Vast amounts of training data is available.*

## Surprising Phenomenon:



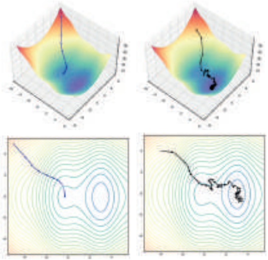


(Source: Belkin, Hsu, Ma, Mandal; 2019)

# Second Appearance of Neural Networks

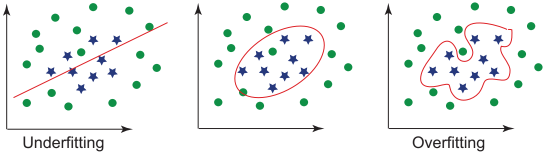
## Key Observations by Y. LeCun et al. (around 2000):

- Drastic improvement of computing power.
  $\rightsquigarrow$ *Networks with hundreds of layers can be trained.*
  $\rightsquigarrow$ *Deep Neural Networks!*
- Age of Data starts.
  $\rightsquigarrow$ *Vast amounts of training data is available.*



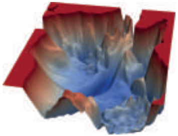
(Source: Berner, Grohs, K, Petersen; 2021)

## Surprising Phenomenon:





(Source: Belkin, Hsu, Ma, Mandal; 2019)



# Mathematics for Deep Learning

- **Expressivity:**
  - Which *aspects of a neural network architecture* affect the performance of deep learning?

$\rightsquigarrow$ *Applied Harmonic Analysis, Approximation Theory, ...*

# Mathematics for Deep Learning

- **Expressivity:**
  - Which *aspects of a neural network architecture* affect the performance of deep learning?

$\rightsquigarrow$ *Applied Harmonic Analysis, Approximation Theory, ...*

- **Learning:**
  - Why does *stochastic gradient descent* converge to good local minima despite the non-convexity of the problem?

$\rightsquigarrow$ *Algebraic/Differential Geometry, Optimal Control, Optimization, ...*

# Mathematics for Deep Learning

- **Expressivity:**
  - Which *aspects of a neural network architecture* affect the performance of deep learning?

$\rightsquigarrow$ *Applied Harmonic Analysis, Approximation Theory, ...*

- **Learning:**
  - Why does *stochastic gradient descent* converge to good local minima despite the non-convexity of the problem?

$\rightsquigarrow$ *Algebraic/Differential Geometry, Optimal Control, Optimization, ...*

- **Generalization:**
  - What is the *role of depth*?
  - Why do large neural networks *not overfit*?

$\rightsquigarrow$ *Learning Theory, Probability Theory, Statistics, ...*

# Mathematics for Deep Learning

- **Expressivity:**
  - Which *aspects of a neural network architecture* affect the performance of deep learning?

  $\rightsquigarrow$ *Applied Harmonic Analysis, Approximation Theory, ...*
- **Learning:**
  - Why does *stochastic gradient descent* converge to good local minima despite the non-convexity of the problem?

  $\rightsquigarrow$ *Algebraic/Differential Geometry, Optimal Control, Optimization, ...*
- **Generalization:**
  - What is the *role of depth*?
  - Why do large neural networks *not overfit*?

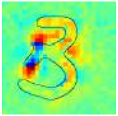
  $\rightsquigarrow$ *Learning Theory, Probability Theory, Statistics, ...*
- **Explainability:**
  - Why did a trained deep neural network *reach a certain decision*?
  - Which *features of data* are learned by deep architectures?

  $\rightsquigarrow$ *Information Theory, Uncertainty Quantification, ...*

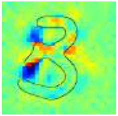
# Explainability

**Main Goal:** We aim to *understand* decisions of "black-box" predictors!

map for digit 3



map for digit 8



**Selected Questions:**

- What *exactly* is relevance in a mathematical sense?
- Can we develop a theory for *optimal relevance maps*?
- How to extend to *challenging modalities*?

Source: **Rate-Distortion Explanation (RDE)**
(Macdonald, Wäldchen, Hauch, K; 2020)
(Heiß, Levie, Resnick, K, Bruna; 2021)

**Vision:**

*Explanation of a decision indistinguishable from a human being!*

$\rightsquigarrow$ *Requires interdisciplinary approach and novel mathematics!*

# Mathematics for Deep Learning

- **Expressivity:**
  - Which *aspects of a neural network architecture* affect the performance of deep learning?

  $\rightsquigarrow$ *Applied Harmonic Analysis, Approximation Theory, ...*
- **Learning:**
  - Why does *stochastic gradient descent* converge to good local minima despite the non-convexity of the problem?

  $\rightsquigarrow$ *Algebraic/Differential Geometry, Optimal Control, Optimization, ...*
- **Generalization:**
  - What is the *role of depth*?
  - Why do large neural networks *not overfit*?

  $\rightsquigarrow$ *Learning Theory, Probability Theory, Statistics, ...*
- **Explainability:**
  - Why did a trained deep neural network *reach a certain decision*?
  - Which *features of data* are learned by deep architectures?

  $\rightsquigarrow$ *Information Theory, Uncertainty Quantification, ...*

# Deep Learning for Mathematics

- **Inverse Problems:**
  - How do we *optimally combine* deep learning with model-based approaches?
  - Are neural networks capable of *replacing highly specialized numerical algorithms* in natural sciences?

$\rightsquigarrow$ *Imaging Science, Inverse Problems, Microlocal Analysis, ...*

# Deep Learning for Mathematics

- **Inverse Problems:**
  - How do we *optimally combine* deep learning with model-based approaches?
  - Are neural networks capable of *replacing highly specialized numerical algorithms* in natural sciences?

$\rightsquigarrow$ *Imaging Science, Inverse Problems, Microlocal Analysis, ...*

- **Partial Differential Equations:**
  - Why do neural networks perform well in *very high-dimensional environments*?
  - Are neural networks capable of *replacing highly specialized numerical algorithms* in natural sciences?

$\rightsquigarrow$ *Numerical Mathematics, Partial Differential Equations, ...*

# Deep Learning for Mathematics

- **Inverse Problems:**
  - How do we *optimally combine* deep learning with model-based approaches?
  - Are neural networks capable of *replacing highly specialized numerical algorithms* in natural sciences?

$\rightsquigarrow$ *Imaging Science, Inverse Problems, Microlocal Analysis, ...*

- **Partial Differential Equations:**
  - Why do neural networks perform well in *very high-dimensional environments*?
  - Are neural networks capable of *replacing highly specialized numerical algorithms* in natural sciences?

$\rightsquigarrow$ *Numerical Mathematics, Partial Differential Equations, ...*

**Are deep neural networks at least as good as all previous mathematical methods?**

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

*Revisiting Classical Approximation Theory*

# Function Approximation in a Nutshell

**Goal:** Given $\mathcal{C} \subseteq L^2(\mathbb{R}^d)$ and $(\varphi_i)_{i\in I} \subseteq L^2(\mathbb{R}^d)$. Measure the suitability of $(\varphi_i)_{i\in I}$ for uniformly approximating functions from $\mathcal{C}$.

**Definition:** The *error of best N-term approximation* of some $f \in \mathcal{C}$ is given by

$$\|f - f_N\|_2 := \inf_{I_N \subset I, \# I_N = N, (c_i)_{i\in I_N}} \|f - \sum_{i \in I_N} c_i \varphi_i\|_2.$$

The largest $\gamma > 0$ such that

$$\sup_{f\in\mathcal{C}} \|f - f_N\|_2 = O(N^{-\gamma}) \qquad \text{as } N \to \infty$$

determines the *optimal (sparse) approximation rate* of $\mathcal{C}$ by $(\varphi_i)_{i\in I}$.

# Function Approximation in a Nutshell

**Goal:** Given $\mathcal{C} \subseteq L^2(\mathbb{R}^d)$ and $(\varphi_i)_{i\in I} \subseteq L^2(\mathbb{R}^d)$. Measure the suitability of $(\varphi_i)_{i\in I}$ for uniformly approximating functions from $\mathcal{C}$.

**Definition:** The *error of best N-term approximation* of some $f \in \mathcal{C}$ is given by

$$\|f - f_N\|_2 := \inf_{I_N \subset I, \# I_N = N, (c_i)_{i\in I_N}} \|f - \sum_{i \in I_N} c_i \varphi_i\|_2.$$

The largest $\gamma > 0$ such that

$$\sup_{f\in\mathcal{C}} \|f - f_N\|_2 = O(N^{-\gamma}) \qquad \text{as } N \to \infty$$

determines the *optimal (sparse) approximation rate* of $\mathcal{C}$ by $(\varphi_i)_{i\in I}$.

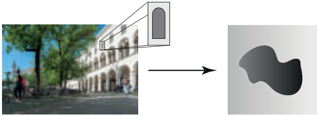
*Approximation accuracy $\leftrightarrow$ Complexity of approximating system in terms of sparsity*

# Modeling Anisotropic Structures

**Definition (Donoho; 2001):**
The set of *cartoon-like functions* $\mathcal{E}^2(\mathbb{R}^2)$ is defined by

$$\mathcal{E}^2(\mathbb{R}^2) = \{f \in L^2(\mathbb{R}^2) : f = f_0 + f_1 \cdot \chi_B\},$$

where $\emptyset \neq B \subset [0,1]^2$ simply connected with $C^2$-boundary and bounded curvature, and $f_i \in C^2(\mathbb{R}^2)$ with supp $f_i \subseteq [0,1]^2$ and $\|f_i\|_{C^2} \leq 1$, $i = 0, 1$.

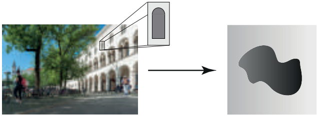


# Modeling Anisotropic Structures

**Definition (Donoho; 2001):**
The set of *cartoon-like functions* $\mathcal{E}^2(\mathbb{R}^2)$ is defined by

$$\mathcal{E}^2(\mathbb{R}^2) = \{f \in L^2(\mathbb{R}^2) : f = f_0 + f_1 \cdot \chi_B\},$$

where $\emptyset \neq B \subset [0,1]^2$ simply connected with $C^2$-boundary and bounded curvature, and $f_i \in C^2(\mathbb{R}^2)$ with supp $f_i \subseteq [0,1]^2$ and $\|f_i\|_{C^2} \leq 1$, $i = 0, 1$.



**Theorem (Donoho; 2001):**
Let $(\psi_\lambda)_\lambda \subseteq L^2(\mathbb{R}^2)$. Allowing only polynomial depth search, we have the following *optimal behavior* for $f \in \mathcal{E}^2(\mathbb{R}^2)$:

$$\|f - f_N\|_2 \asymp N^{-1} \quad \text{as } N \to \infty.$$

# What can Wavelets do?

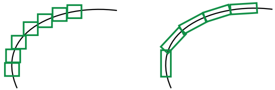
**Problem:**

- *Isotropic* structure of wavelets:

$$\{2^j \psi(\begin{pmatrix} 2^j & 0 \\ 0 & 2^j \end{pmatrix} x - m) : j \in \mathbb{Z}, m \in \mathbb{Z}^2\}, \quad \psi \in L^2(\mathbb{R}^2).$$

- For $f \in \mathcal{E}^2(\mathbb{R}^2)$, wavelets *only* achieve

$$\|f - f_N\|_2 \asymp N^{-\frac{1}{2}}, \quad N \to \infty.$$



# What can Wavelets do?

## Problem:

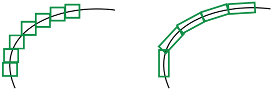
- *Isotropic* structure of wavelets:

$$\{2^j \psi(\begin{pmatrix} 2^j & 0 \\ 0 & 2^j \end{pmatrix} x - m) : j \in \mathbb{Z}, m \in \mathbb{Z}^2\}, \quad \psi \in L^2(\mathbb{R}^2).$$

- For $f \in \mathcal{E}^2(\mathbb{R}^2)$, wavelets *only* achieve

$$\|f - f_N\|_2 \asymp N^{-\frac{1}{2}}, \quad N \to \infty.$$

## Non-Exhaustive List of Approaches:

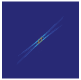


- Ridgelets (Candès and Donoho; 1999)
- Curvelets (Candès and Donoho; 2002)
- Contourlets (Do and Vetterli; 2002)
- Bandlets (LePennec and Mallat; 2003)
- *Shearlets* (K and Labate; 2006)

# (Cone-adapted) Discrete Shearlet Systems

**Parabolic scaling ('width ≈ length$^2$'):**

$$A_{2^j} = \begin{pmatrix} 2^j & 0 \\ 0 & 2^{j/2} \end{pmatrix}, \quad j \in \mathbb{Z}.$$



**Orientation via shearing:**

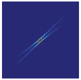
$$S_k = \begin{pmatrix} 1 & k \\ 0 & 1 \end{pmatrix}, \quad k \in \mathbb{Z}.$$

# (Cone-adapted) Discrete Shearlet Systems

**Parabolic scaling ('width $\approx$ length$^2$'):**

$$A_{2^j} = \begin{pmatrix} 2^j & 0 \\ 0 & 2^{j/2} \end{pmatrix}, \quad j \in \mathbb{Z}.$$

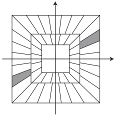
**Orientation via shearing:**



$$S_k = \begin{pmatrix} 1 & k \\ 0 & 1 \end{pmatrix}, \quad k \in \mathbb{Z}.$$

**Definition (K, Labate; 2006):**

The *(cone-adapted) discrete shearlet system* $\mathcal{SH}(\phi, \psi, \tilde{\psi})$ generated by $\phi \in L^2(\mathbb{R}^2)$ and $\psi, \tilde{\psi} \in L^2(\mathbb{R}^2)$ is the union of



$$\{\phi(\cdot - m) : m \in \mathbb{Z}^2\},$$

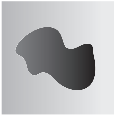
$$\{2^{3j/4}\psi(S_k A_{2^j} \cdot - m) : j \geq 0, |k| \leq \lceil 2^{j/2} \rceil, m \in \mathbb{Z}^2\},$$

$$\{2^{3j/4}\tilde{\psi}(\tilde{S}_k \tilde{A}_{2^j} \cdot - m) : j \geq 0, |k| \leq \lceil 2^{j/2} \rceil, m \in \mathbb{Z}^2\}.$$

The associated *shearlet transform* will be denoted by SH.

**Theorem (K, Lim; 2011):**

Let $\phi, \psi, \tilde{\psi} \in L^2(\mathbb{R}^2)$ be compactly supported, and let $\hat{\psi}$, $\hat{\tilde{\psi}}$ satisfy certain decay condition. Then $\mathcal{SH}(\phi, \psi, \tilde{\psi})$ provides an *optimally sparse approximation* of $f \in \mathcal{E}^2(\mathbb{R}^2)$, i.e.,



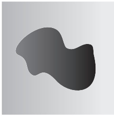
$$\|f - f_N\|_2 \lesssim N^{-1}(\log N)^{\frac{3}{2}} \quad \text{as } N \to \infty.$$

# Optimally Sparse Approximation

**Theorem (K, Lim; 2011):**

Let $\phi, \psi, \tilde{\psi} \in L^2(\mathbb{R}^2)$ be compactly supported, and let $\hat{\psi}$, $\hat{\tilde{\psi}}$ satisfy certain decay condition. Then $\mathcal{SH}(\phi, \psi, \tilde{\psi})$ provides an *optimally sparse approximation* of $f \in \mathcal{E}^2(\mathbb{R}^2)$, i.e.,

$$\|f - f_N\|_2 \lesssim N^{-1}(\log N)^{\frac{3}{2}} \quad \text{as } N \to \infty.$$



**2D&3D (parallelized) Fast Shearlet Transform (`www.ShearLab.org`):**

- Matlab *(K, Lim, Reisenhofer; 2013)*
- Julia *(Loarca; 2017)*
- Python *(Look; 2018)*
- Tensorflow *(K, Loarca; 2019)*

# Function Approximation in a Nutshell

**Goal:** Given $\mathcal{C} \subseteq L^2(\mathbb{R}^d)$ and $(\varphi_i)_{i\in I} \subseteq L^2(\mathbb{R}^d)$. Measure the suitability of $(\varphi_i)_{i\in I}$ for uniformly approximating functions from $\mathcal{C}$.

**Definition:** The *error of best $N$-term approximation* of some $f \in \mathcal{C}$ is given by

$$\|f - f_N\|_2 := \inf_{I_N \subset I, \# I_N = N, (c_i)_{i\in I_N}} \|f - \sum_{i\in I_N} c_i \varphi_i\|_2.$$

The largest $\gamma > 0$ such that

$$\sup_{f\in\mathcal{C}} \|f - f_N\|_2 = O(N^{-\gamma}) \qquad \text{as } N \to \infty$$

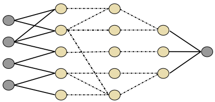
determines the *optimal (sparse) approximation rate* of $\mathcal{C}$ by $(\varphi_i)_{i\in I}$.

*Approximation accuracy $\leftrightarrow$ Complexity of approximating system*
*in terms of sparsity*

*Universality of Deep Neural Networks:*

*An Analysis of Their Expressivity*

# Complexity of a Deep Neural Network



**Recall:**

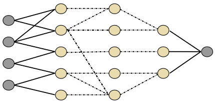
- $L$: Number of layers.
- $\rho : \mathbb{R} \to \mathbb{R}$: *Activation function*.
- $T_\ell : \mathbb{R}^{N_{\ell-1}} \to \mathbb{R}^{N_\ell}$, $\ell = 1, \dots, L$, where $T_\ell x = W^{(\ell)} x + b^{(\ell)}$

Then $\Phi : \mathbb{R}^d \to \mathbb{R}^{N_L}$ given by

$$\Phi(x) = T_L \rho(T_{L-1} \rho(\dots \rho(T_1(x))), \quad x \in \mathbb{R}^d,$$

is called *(deep) neural network (DNN)*.

# Complexity of a Deep Neural Network



**Recall:**

- $L$: Number of layers.
- $\rho : \mathbb{R} \to \mathbb{R}$: *Activation function*.
- $T_\ell : \mathbb{R}^{N_{\ell-1}} \to \mathbb{R}^{N_\ell}$, $\ell = 1, \ldots, L$, where $T_\ell x = W^{(\ell)} x + b^{(\ell)}$

Then $\Phi : \mathbb{R}^d \to \mathbb{R}^{N_L}$ given by

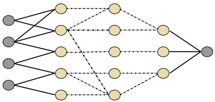
$$\Phi(x) = T_L \rho(T_{L-1} \rho(\ldots \rho(T_1(x))), \quad x \in \mathbb{R}^d,$$

is called *(deep) neural network (DNN)*.

**Measure for Complexity:** The *complexity $C(\Phi)$* is defined by

$$C(\Phi) := \sum_{\ell=1}^{L} \left( \|W^{(\ell)}\|_0 + \|b^{(\ell)}\|_0 \right).$$

# Complexity of a Deep Neural Network



**Recall:**

- $L$: Number of layers.
- $\rho : \mathbb{R} \to \mathbb{R}$: *Activation function*.
- $T_\ell : \mathbb{R}^{N_{\ell-1}} \to \mathbb{R}^{N_\ell}$, $\ell = 1, \dots, L$, where $T_\ell x = W^{(\ell)} x + b^{(\ell)}$

Then $\Phi : \mathbb{R}^d \to \mathbb{R}^{N_L}$ given by

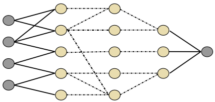
$$\Phi(x) = T_L \rho(T_{L-1} \rho(\dots \rho(T_1(x)))), \quad x \in \mathbb{R}^d,$$

is called *(deep) neural network (DNN)*. We write $\Phi \in \mathcal{NN}_{L,C(\Phi),d,\rho}$.

**Measure for Complexity:** The *complexity* $C(\Phi)$ is defined by

$$C(\Phi) := \sum_{\ell=1}^{L} \left( \| W^{(\ell)} \|_0 + \| b^{(\ell)} \|_0 \right).$$

# Complexity of a Deep Neural Network



**Recall:**

- $L$: Number of layers.
- $\rho : \mathbb{R} \to \mathbb{R}$: *Activation function*.
- $T_\ell : \mathbb{R}^{N_{\ell-1}} \to \mathbb{R}^{N_\ell}$, $\ell = 1, \ldots, L$, where $T_\ell x = W^{(\ell)} x + b^{(\ell)}$

Then $\Phi : \mathbb{R}^d \to \mathbb{R}^{N_L}$ given by

$$\Phi(x) = T_L \rho(T_{L-1} \rho(\ldots \rho(T_1(x)))), \quad x \in \mathbb{R}^d,$$

is called *(deep) neural network (DNN)*. We write $\Phi \in \mathcal{NN}_{L,C(\Phi),d,\rho}$.

**Measure for Complexity:** The *complexity* $C(\Phi)$ is defined by

$$C(\Phi) := \sum_{\ell=1}^{L} \left( \|W^{(\ell)}\|_0 + \|b^{(\ell)}\|_0 \right).$$

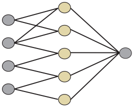
**Key Challenge:**

*Approximation accuracy* $\leftrightarrow$ *Complexity of approximating network in terms of memory efficiency!*

# One Size Fits All?

**Universal Approximation Theorem (Cybenko, 1989)(Hornik, 1991):**
Let $d \in \mathbb{N}$, $K \subset \mathbb{R}^d$ compact, $f : K \to \mathbb{R}$ continuous, $\rho : \mathbb{R} \to \mathbb{R}$ continuous and not a polynomial. Then, for each $\epsilon > 0$, there exist $N \in \mathbb{N}$, $a_k, b_k \in \mathbb{R}, w_k \in \mathbb{R}^d$ such that

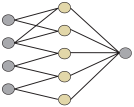
$$\|f - \sum_{k=1}^{N} a_k \rho(\langle w_k, \cdot \rangle - b_k)\|_\infty \leq \epsilon.$$



# One Size Fits All?

**Universal Approximation Theorem (Cybenko, 1989)(Hornik, 1991):**
Let $d \in \mathbb{N}$, $K \subset \mathbb{R}^d$ compact, $f : K \to \mathbb{R}$ continuous, $\rho : \mathbb{R} \to \mathbb{R}$ continuous and not a polynomial. Then, for each $\epsilon > 0$, there exist $N \in \mathbb{N}$, $a_k, b_k \in \mathbb{R}, w_k \in \mathbb{R}^d$ such that

$$\|f - \sum_{k=1}^{N} a_k \rho(\langle w_k, \cdot\rangle - b_k)\|_\infty \leq \epsilon.$$

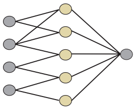


*The complexity can be arbitrarily large!*

# One Size Fits All?

**Universal Approximation Theorem (Cybenko, 1989)(Hornik, 1991):** Let $d \in \mathbb{N}$, $K \subset \mathbb{R}^d$ compact, $f : K \to \mathbb{R}$ continuous, $\rho : \mathbb{R} \to \mathbb{R}$ continuous and not a polynomial. Then, for each $\epsilon > 0$, there exist $N \in \mathbb{N}$, $a_k, b_k \in \mathbb{R}, w_k \in \mathbb{R}^d$ such that

$$\|f - \sum_{k=1}^{N} a_k \rho(\langle w_k, \cdot \rangle - b_k)\|_\infty \leq \epsilon.$$



*The complexity can be arbitrarily large!*

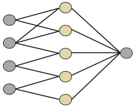
**Theorem (Yarotsky; 2017):** For all $f \in \mathcal{C} = C^s([0,1]^d)$ and $\rho$ the ReLU, i.e., $\rho(x) = \max\{0, x\}$, there exist neural networks $(\Phi_n)_{n \in \mathbb{N}}$ with $L(\Phi_n) \approx \log(n)$ such that

$$\|f - \Phi_n\|_\infty \lesssim C(\Phi_n)^{-\frac{s}{d}} \to 0 \quad \text{as } n \to \infty.$$

# One Size Fits All?

**Universal Approximation Theorem (Cybenko, 1989)(Hornik, 1991):** Let $d \in \mathbb{N}$, $K \subset \mathbb{R}^d$ compact, $f : K \to \mathbb{R}$ continuous, $\rho : \mathbb{R} \to \mathbb{R}$ continuous and not a polynomial. Then, for each $\epsilon > 0$, there exist $N \in \mathbb{N}$, $a_k, b_k \in \mathbb{R}, w_k \in \mathbb{R}^d$ such that

$$\|f - \sum_{k=1}^{N} a_k \rho(\langle w_k, \cdot \rangle - b_k)\|_\infty \leq \epsilon.$$



*The complexity can be arbitrarily large!*

**Theorem (Yarotsky; 2017):** For all $f \in \mathcal{C} = C^s([0,1]^d)$ and $\rho$ the ReLU, i.e., $\rho(x) = \max\{0, x\}$, there exist neural networks $(\Phi_n)_{n \in \mathbb{N}}$ with $L(\Phi_n) \approx \log(n)$ such that

$$\|f - \Phi_n\|_\infty \lesssim C(\Phi_n)^{-\frac{s}{d}} \to 0 \quad \text{as } n \to \infty.$$

*This result is not optimal!*

# A Fundamental Lower Bound

**Complexity of a Function Class:**

The *optimal exponent* $\gamma^*(\mathcal{C})$ measures the complexity of $\mathcal{C} \subset L^2(\mathbb{R}^d)$.

# A Fundamental Lower Bound

**Complexity of a Function Class:**
The *optimal exponent* $\gamma^*(\mathcal{C})$ measures the complexity of $\mathcal{C} \subset L^2(\mathbb{R}^d)$.

**Theorem (Bölcskei, Grohs, K, and Petersen; 2019):**
Let $d \in \mathbb{N}$, $\rho : \mathbb{R} \to \mathbb{R}$, and let $\mathcal{C} \subset L^2(\mathbb{R}^d)$. Further, let

$$\textbf{Learn} : (0,1) \times \mathcal{C} \to \mathcal{NN}_{\infty,\infty,d,\rho}$$

satisfy that, for each $f \in \mathcal{C}$ and $0 < \epsilon < 1$,

$$\sup_{f \in \mathcal{C}} \|f - \textbf{Learn}(\epsilon, f)\|_2 \leq \epsilon.$$

Then, for all $\gamma < \gamma^*(\mathcal{C})$,

$$\epsilon^\gamma \sup_{f \in \mathcal{C}} C(\textbf{Learn}(\epsilon, f)) \to \infty, \qquad \text{as } \epsilon \to 0.$$

*Conceptual bound independent on the learning algorithm!*

# A Fundamental Lower Bound

**Complexity of a Function Class:**

The *optimal exponent* $\gamma^*(\mathcal{C})$ measures the complexity of $\mathcal{C} \subset L^2(\mathbb{R}^d)$.

**Theorem (Bölcskei, Grohs, K, and Petersen; 2019):**

Let $d \in \mathbb{N}$, $\rho : \mathbb{R} \to \mathbb{R}$, and let $\mathcal{C} \subset L^2(\mathbb{R}^d)$. Further, let

$$\mathbf{Learn} : (0,1) \times \mathcal{C} \to \mathcal{NN}_{\infty,\infty,d,\rho}$$

satisfy that, for each $f \in \mathcal{C}$ and $0 < \epsilon < 1$,

$$\sup_{f \in \mathcal{C}} \| f - \mathbf{Learn}(\epsilon, f) \|_2 \leq \epsilon.$$

Then, for all $\gamma < \gamma^*(\mathcal{C})$,

$$\epsilon^{\gamma} \sup_{f \in \mathcal{C}} C(\mathbf{Learn}(\epsilon, f)) \to \infty, \qquad \text{as } \epsilon \to 0.$$

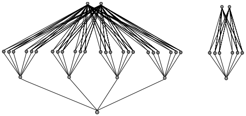
*Conceptual bound independent on the learning algorithm!*

$\rightsquigarrow$ *What happens for $\gamma = \gamma^*(\mathcal{C})$?*

# Optimal Approximation

**Key Ideas for a Specific Function Class:**

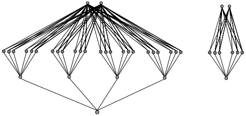
- Consider a representation system with an optimal approximation rate.
- Realize each element of a representation system by a neural network.
- Mimic best $N$-term approximation by networks.



# Optimal Approximation

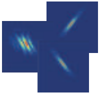
**Key Ideas for a Specific Function Class:**

- Consider a representation system with an optimal approximation rate.
- Realize each element of a representation system by a neural network.
- Mimic best $N$-term approximation by networks.



**Choice for our Result:**

Use the affine system of *shearlets*.



**Theorem (Bölcskei, Grohs, K, and Petersen; 2019):**

Let $\rho$ be a suitably chosen, and let $\epsilon > 0$. Then, for all $f \in \mathcal{E}^2(\mathbb{R}^2)$ and $N \in \mathbb{N}$, there exists $\Phi \in \mathcal{NN}_{3,O(N),2,\rho}$ with

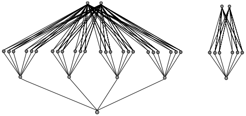
$$\|f - \Phi\|_2 \lesssim N^{-1+\epsilon} \to 0 \quad \text{as } N \to \infty.$$

*This is the optimal rate; hence the first bound is sharp!*

# Optimal Approximation

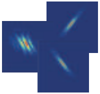
## Key Ideas for a Specific Function Class:

- Consider a representation system with an optimal approximation rate.
- Realize each element of a representation system by a neural network.
- Mimic best $N$-term approximation by networks.



## Choice for our Result:

Use the affine system of *shearlets*.



## Theorem (Bölcskei, Grohs, K, and Petersen; 2019):

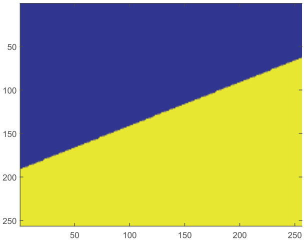
Let $\rho$ be a suitably chosen, and let $\epsilon > 0$. Then, for all $f \in \mathcal{E}^2(\mathbb{R}^2)$ and $N \in \mathbb{N}$, there exists $\Phi \in \mathcal{NN}_{3,O(N),2,\rho}$ with

$$\|f - \Phi\|_2 \lesssim N^{-1+\epsilon} \to 0 \quad \text{as } N \to \infty.$$

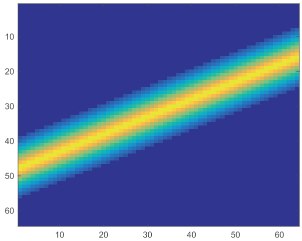
*This is the optimal rate; hence the first bound is sharp!*

**Deep neural networks achieve optimal approximation properties of all affine systems combined!**

# Numerical Experiments (with ReLUs & Backpropagation)

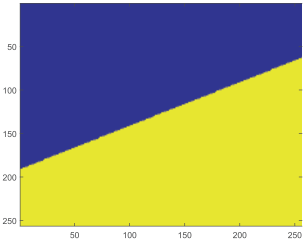



Linear Singularity

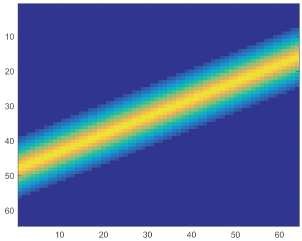



Subnetworks: $\approx$ *Ridgelets!*

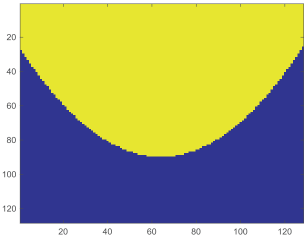
# Numerical Experiments (with ReLUs & Backpropagation)



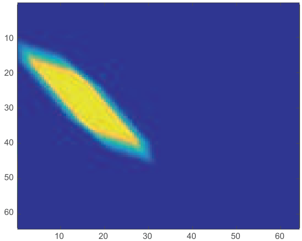

Linear Singularity




Subnetworks: $\approx$ *Ridgelets!*




Curvilinear Singularity




Subnetworks: $\approx$ *Shearlets!*

*Are Deep Neural Networks Really Better Than Classical Methods?*

# Solving Inverse Problems

**Sparse Regularization:**

Given an *(ill-posed) inverse problem*

$$Kf = g, \qquad \text{where} \quad K : X \to Y,$$

an approximate solution $f^\alpha \in X$, $\alpha > 0$, can be determined by

$$f^\alpha := \operatorname*{argmin}_f \Big[ \underbrace{\|Kf - g\|^2}_{\text{Data fidelity term}} + \alpha \cdot \underbrace{\|(\langle f, \varphi_i \rangle)_{i \in I}\|_1}_{\text{Penalty term}} \Big].$$

# Solving Inverse Problems

**Sparse Regularization:**

Given an *(ill-posed) inverse problem*

$$Kf = g, \qquad \text{where} \quad K : X \to Y,$$

an approximate solution $f^\alpha \in X$, $\alpha > 0$, can be determined by

$$f^\alpha := \operatorname*{argmin}_f \Big[ \underbrace{\|Kf - g\|^2}_{\text{Data fidelity term}} + \alpha \cdot \underbrace{\|(\langle f, \varphi_i \rangle)_{i\in I}\|_1}_{\text{Penalty term}} \Big].$$

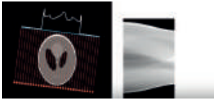
**Some Typical Deep Learning Approaches to Inverse Problems:**

Iterative solvers, e.g., ADMM, contain a ...

- *denoising step*, which can be replaced by a neural network.
  $\rightsquigarrow$ *Plug-and-play with CNN-denoising* [Venkatakrishnan,Bouman,Wohlberg,'13], [Romano,Elad,Milanfar,'16], [Meinhardt et al.,'17], [Reehorst,Schniter,'19] ...
- *proximal steps*, which can be learnt using a deep learning-based approach.
  $\rightsquigarrow$ *Learned Iterative Schemes* [Gregor,LeCun,'10], [Yang et al.,'16], [Hammernick et al.,'16] [Adler,Öktem,'17], [Hammernick et al.,'18], [Hauptmann et al.,'18] ...

# (Limited Angle-) Computed Tomography

A CT scanner samples the *Radon transform*

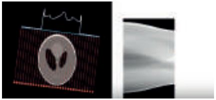


$$\mathcal{R}f(\phi, s) = \int_{L(\phi,s)} f(x) dS(x),$$

for $L(\phi, s) = \left\{x \in \mathbb{R}^2 : x_1 \cos(\phi) + x_2 \sin(\phi) = s\right\}$, $\phi \in [-\pi/2, \pi/2)$, and $s \in \mathbb{R}$.

# (Limited Angle-) Computed Tomography

A CT scanner samples the *Radon transform*



$$\mathcal{R}f(\phi, s) = \int_{L(\phi,s)} f(x) dS(x),$$

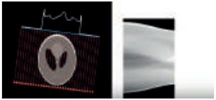
for $L(\phi, s) = \{x \in \mathbb{R}^2 : x_1 \cos(\phi) + x_2 \sin(\phi) = s\}$, $\phi \in [-\pi/2, \pi/2)$, and $s \in \mathbb{R}$.

Challenging inverse problem if $\mathcal{R}f(\cdot, s)$ is only sampled on $[-\phi, \phi] \subset [-\pi/2, \pi/2)$.

**Applications:** Dental CT, electron tomography,...

# (Limited Angle-) Computed Tomography

A CT scanner samples the *Radon transform*



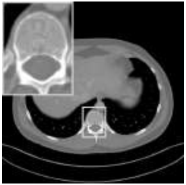
$$\mathcal{R}f(\phi, s) = \int_{L(\phi,s)} f(x)dS(x),$$

for $L(\phi, s) = \{x \in \mathbb{R}^2 : x_1 \cos(\phi) + x_2 \sin(\phi) = s\}$, $\phi \in [-\pi/2, \pi/2)$, and $s \in \mathbb{R}$.

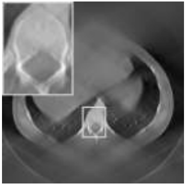
> Challenging inverse problem if $\mathcal{R}f(\cdot, s)$ is only sampled on $[-\phi, \phi] \subset [-\pi/2, \pi/2)$.

**Applications:** Dental CT, electron tomography,...

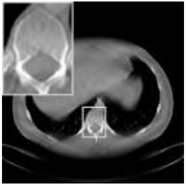
**Model-Based Approaches Fail ($60^\circ$ Missing Angle):**



Original Image

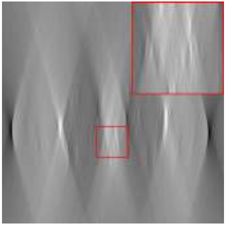


Filtered Backprojection



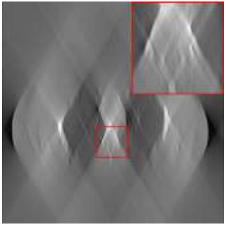
Sparse Regularization with Shearlets

# Zooming in on the Limited-Angle CT Problem



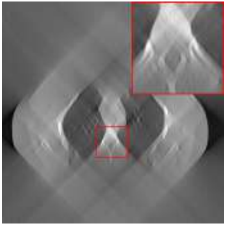
$\phi = 15°$, filtered backprojection (FBP)

# Zooming in on the Limited-Angle CT Problem



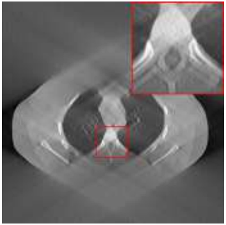
$\phi = 30^\circ$, filtered backprojection (FBP)

# Zooming in on the Limited-Angle CT Problem



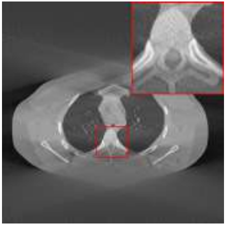
$\phi = 45°$, filtered backprojection (FBP)

# Zooming in on the Limited-Angle CT Problem



$\phi = 60^\circ$, filtered backprojection (FBP)

# Zooming in on the Limited-Angle CT Problem



$\phi = 75^\circ$, filtered backprojection (FBP)

# Zooming in on the Limited-Angle CT Problem

$\phi = 90^\circ$, filtered backprojection (FBP)

# Zooming in on the Limited-Angle CT Problem

$\phi = 90^\circ$, filtered backprojection (FBP)

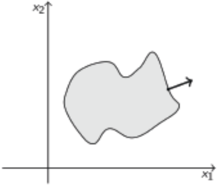
## Illustration of Theorem [Quinto, 1993]:

*'visible"*: singularities tangent to sampled lines

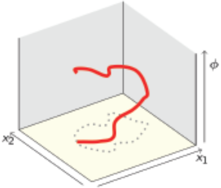
*"invisible"*: singularities not tangent to sampled lines

# Shearlets can Help

**Key Idea:** Filling the missing angle is an inpainting problem of the wavefront set!

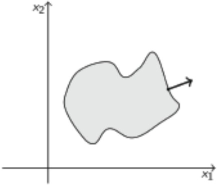



$f = 1_D$ for a set $D \subseteq \mathbb{R}^2$
with smooth boundary

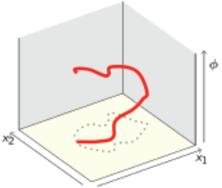


# Shearlets can Help

**Key Idea:** Filling the missing angle is an inpainting problem of the wavefront set!



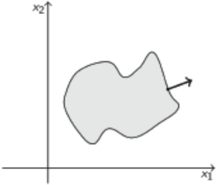

$f = 1_D$ for a set $D \subseteq \mathbb{R}^2$
with smooth boundary



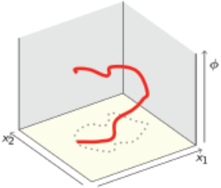

Theorem (K, Labate; 2006):
"Shearlets can identify the wavefront set at fine scales."

# Shearlets can Help

**Key Idea:** Filling the missing angle is an inpainting problem of the wavefront set!

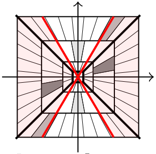



$f = 1_D$ for a set $D \subseteq \mathbb{R}^2$
with smooth boundary




Theorem (K, Labate; 2006):
"Shearlets can identify the wavefront set at fine scales."

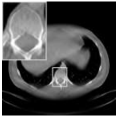
Shearlets can Separate the Visible and Invisible Part:



# Our Approach "Learn the Invisible (LtI)"

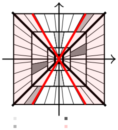
(Bubba, K, Lassas, März, Samek, Siltanen, Srinivan; 2019)

**Step 1:** *Reconstruct the visible*



$$f^* := \underset{f \geq 0}{\operatorname{argmin}} \| \mathcal{R}_\phi f - g\|_2^2 + \| \mathrm{SH}_\psi(f)\|_{1,w}$$

- Best available classical solution (little artifacts, denoised)
- Access "wavefront set" via sparsity prior on shearlets:
  - For $(j,k,l) \in \mathcal{I}_{\mathtt{inv}}$: $\mathrm{SH}_\psi(f^*)_{(j,k,l)} \approx 0$
  - For $(j,k,l) \in \mathcal{I}_{\mathtt{vis}}$: $\mathrm{SH}_\psi(f^*)_{(j,k,l)}$ reliable and near perfect



**Step 2:** *Learn the invisible*

$$\mathcal{NN}_\theta : \ \mathrm{SH}_\psi(f^*)_{\mathcal{I}_{\mathtt{vis}}} \longrightarrow F \ \left( \overset{!}{\approx} \mathrm{SH}_\psi(f_{\mathrm{gt}})_{\mathcal{I}_{\mathtt{inv}}} \right)$$

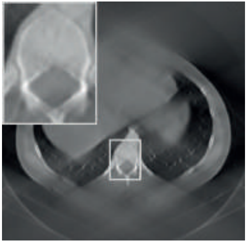

U-Net

**Step 3:** *Combine*

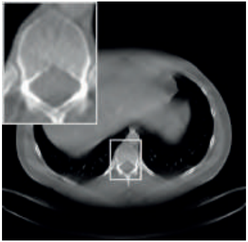
$$f_{\mathtt{LtI}} = \mathrm{SH}_\psi^T \left(\mathrm{SH}_\psi(f^*)_{\mathcal{I}_{\mathtt{vis}}} + F\right)$$

# Numerical Results

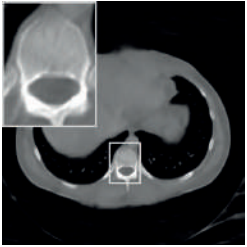
Original



Filtered Backprojection



Sparse Regularization with Shearlets

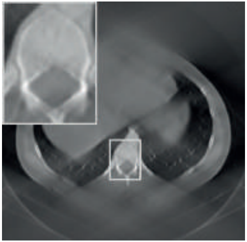


[Gu & Ye, 2017]

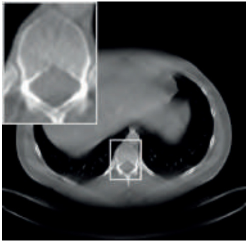
Learn the Invisible (LtI)

# Numerical Results

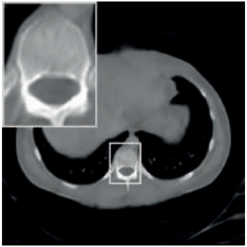
Original



Filtered Backprojection



Sparse Regularization with Shearlets



[Gu & Ye, 2017]

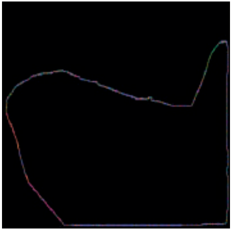
Learn the Invisible (LtI)

*Deep neural networks can outperform classical methods by far!*

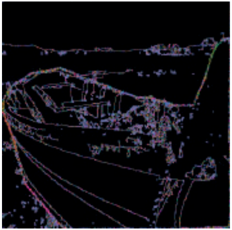
# Deep Network Shearlet Edge Extractor (DeNSE)

(Andrade-Loarca, K, Öktem, Petersen; 2019)

Original

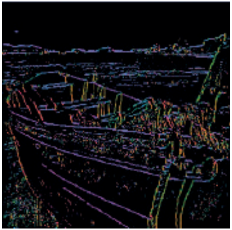


Human Annotation



SEAL [Yu et al; 2018]

CoShREM [Reisenhofer et al.; 2015]



DeNSE

# Deep Learning for Mathematics

- **Inverse Problems:**
  - How do we *optimally combine* deep learning with model-based approaches?
  - Are neural networks capable of *replacing highly specialized numerical algorithms* in natural sciences?

  $\rightsquigarrow$ *Imaging Science, Inverse Problems, Microlocal Analysis, ...*

- **Partial Differential Equations:**
  - Why do neural networks perform well in *very high-dimensional environments*?
  - Are neural networks capable of *replacing highly specialized numerical algorithms* in natural sciences?

  $\rightsquigarrow$ *Numerical Mathematics, Partial Differential Equations, ...*

# Deep Learning for Mathematics

- **Inverse Problems:**
  - How do we *optimally combine* deep learning with model-based approaches?
  - Are neural networks capable of *replacing highly specialized numerical algorithms* in natural sciences?

$\rightsquigarrow$ *Imaging Science, Inverse Problems, Microlocal Analysis, ...*

- **Partial Differential Equations:**
  - Why do neural networks perform well in *very high-dimensional environments*?
  - Are neural networks capable of *replacing highly specialized numerical algorithms* in natural sciences?

$\rightsquigarrow$ *Numerical Mathematics, Partial Differential Equations, ...*

**Why should one use deep neural networks for solving PDEs at all?**

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

*A Final Glimpse into the Effectivness of*

*Deep Neural Networks for Solving PDEs!*

# Numerical Deep Learning Approaches to PDEs

**Common Approach to Solve PDEs with Neural Networks:** Approximate the solution $u$ of a PDE $\mathcal{L}(u) = f$ by a neural network $\Phi$, i.e., determine

$$\mathcal{L}(\Phi) \approx f.$$

**Incomplete List of Contributions:** [Lagaris, Likas, Fotiadis; 1998], [E, Yu; 2017], [Czarnecki, Osindero, Jaderberg, Swirszcz, Pascanu; 2017], [Sirignano, Spiliopoulos; 2017], [Han, Jentzen, E; 2017], [Schwab, Zech; 2019], [Raissi, Perdikaris, Karniadakis; 2020], [Grohs, Herrmann; 2021], ...

# Numerical Deep Learning Approaches to PDEs

**Common Approach to Solve PDEs with Neural Networks:** Approximate the solution $u$ of a PDE $\mathcal{L}(u) = f$ by a neural network $\Phi$, i.e., determine

$$\mathcal{L}(\Phi) \approx f.$$

**Incomplete List of Contributions:** [Lagaris, Likas, Fotiadis; 1998], [E, Yu; 2017], [Czarnecki, Osindero, Jaderberg, Swirszcz, Pascanu; 2017], [Sirignano, Spiliopoulos; 2017], [Han, Jentzen, E; 2017], [Schwab, Zech; 2019], [Raissi, Perdikaris, Karniadakis; 2020], [Grohs, Herrmann; 2021], ...

**Parametric PDEs:** *Parameter dependent families of PDEs* arise in basically any branch of science and engineering:

- Complex design problems
- Optimization tasks
- Uncertainty quantification
- ...

**Parametric Map:**

$$\mathcal{Y} \ni y \;\mapsto\; u_y \in \mathcal{H} \quad \text{such that} \quad \mathcal{L}(u_y, y) = f_y.$$

*Curse of Dimensionality: Computational cost too high!*

# What can Deep Neural Networks do?

**Parametric Map:**

$$\mathbb{R}^p \supseteq \mathcal{Y} \ni y \ \mapsto \ \mathbf{u}_y^{\mathrm{h}} \in \mathbb{R}^D \quad \text{such that} \quad b_y\left(u_y^h, v\right) = f_y(v) \ \text{for all } v.$$

*Can a neural network approximate the parametric map?*

# What can Deep Neural Networks do?

**Parametric Map:**

$$\mathbb{R}^p \supseteq \mathcal{Y} \ni y \mapsto \mathbf{u}_y^{\mathrm{h}} \in \mathbb{R}^D \quad \text{such that} \quad b_y\left(u_y^h, v\right) = f_y(v) \text{ for all } v.$$

*Can a neural network approximate the parametric map?*

**Advantages:**

- After training, extremely rapid computation of the map.
- Flexible, universal approach.

**Questions:** Let $\epsilon > 0$.

(1) Does *there exist a neural network* $\Phi$ such that

$$\|\Phi - \mathbf{u}_y^{\mathrm{h}}\| \leq \epsilon \qquad \text{for all } y \in \mathcal{Y}?$$

(2) How does the *complexity of* $\Phi$ depend on $p$ and $D$?

(3) How do neural networks *perform numerically* on this task?

# Theoretical Results

**Theoretical Approach (K, Petersen, Raslan, Schneider; 2021):**

- There exists a neural network $\Phi$ which approximates the parametric map:
$$\|\Phi - \mathbf{u}_y^{\mathrm{h}}\| \leq \epsilon \qquad \text{for all } y \in \mathcal{Y}.$$
- The dependence of $C(\Phi)$ on $p$ and $D$ can be (polynomially) controlled.

# Theoretical Results

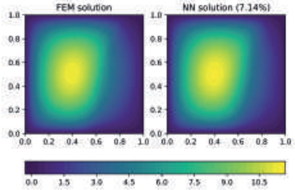
**Theoretical Approach (K, Petersen, Raslan, Schneider; 2021):**

- There exists a neural network $\Phi$ which approximates the parametric map:
$$\|\Phi - \mathbf{u}_y^{\mathrm{h}}\| \leq \epsilon \qquad \text{for all } y \in \mathcal{Y}.$$
- The dependence of $C(\Phi)$ on $p$ and $D$ can be (polynomially) controlled.

**Numerical Results (Geist, Petersen, Raslan, Schneider, K; 2021)**

- Parametric diffusion equation with various parametrizations
- *Fixed neural network architecture*: 11 layers and 0.2-LReLU
- Training set: 20000 i.i.d. parameter samples

**Example ($p = 91$):**




*This performance does also not suffer from the curse of dimensionality!*

*Some Final Thoughts...*

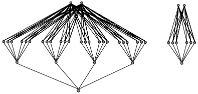
# Conclusions

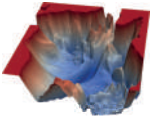
## Deep Learning:

- *Impressive performance* in real-world applications!
- A *theoretical foundation* of is largely missing!

> *(New) Mathematics is crucially needed (...which concerns almost all areas)!*

## Mathematics for Deep Learning:





- *Expressivity*: Optimal architectures?
- *Learning*: Controllable, efficient algorithms?
- *Generalization*: Performance on test data sets?
- *Explainability*: Explaining network decisions?

## Deep Learning for Mathematics:

- Significantly better solvers of *inverse problems*.
- Beating the curse of dimensionality for *partial differential equations*.

# The 7 Mathematical Key Problems of Deep Learning

(1) What is the *role of depth*?

(2) Which *aspects of a neural network architecture* affect the performance of deep learning?

(3) Why does *stochastic gradient descent* converge to good local minima despite the non-convexity of the problem?

(4) Why do large neural networks *not overfit*?

(5) Why do neural networks perform well in *very high-dimensional environments*?

(6) Which *features of data* are learned by deep architectures?

(7) Are neural networks capable of *replacing highly specialized numerical algorithms* in natural sciences?

Source: Berner, Grohs, K, Petersen, The Modern Mathematics of Deep Learning, arXiv:2105.04026.

# The 7 Mathematical Key Problems of Deep Learning

(1) What is the *role of depth*?

(2) Which *aspects of a neural network architecture* affect the performance of deep learning?

(3) Why does *stochastic gradient descent* converge to good local minima despite the non-convexity of the problem?

(4) Why do large neural networks *not overfit*?

(5) Why do neural networks perform well in *very high-dimensional environments*?

(6) Which *features of data* are learned by deep architectures?

(7) Are neural networks capable of *replacing highly specialized numerical algorithms* in natural sciences?

*Exciting Future Perspectives for Mathematics!*

Source: Berner, Grohs, K, Petersen, The Modern Mathematics of Deep Learning, arXiv:2105.04026.

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

FACULTY OF MATHEMATICS, INFORMATICS AND STATISTICS
DEPARTMENT OF MATHEMATICS
MATHEMATICAL FOUNDATIONS OF ARTIFICIAL INTELLIGENCE

# THANK YOU!

**References available at:**

`www.ai.math.lmu.de/kutyniok`

**Survey Paper (arXiv:2105.04026):**

Berner, Grohs, K, Petersen, *The Modern Mathematics of Deep Learning*.

**Check related information on Twitter at:**

`@GittaKutyniok`

**Upcoming Book:**

- P. Grohs and G. Kutyniok
  *Mathematical Aspects of Deep Learning*
  Cambridge University Press (in preparation)