

High-Dimensional Approximation in AI

Lecture 1: Expressivity of Deep Neural Networks

Gitta Kutyniok (LMU Munich)

also

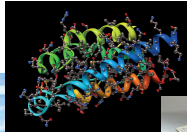
University of Tromsø & DLR – German Aerospace Center

CIME School on High-Dimensional Approximation
Cetraro, Italy, September 22 –27, 2024

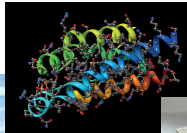
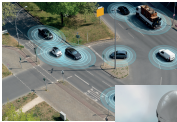


Bavarian AI Chair for
Mathematical Foundations
of Artificial Intelligence

Fourth Industrial Revolution by Artificial Intelligence



Fourth Industrial Revolution by Artificial Intelligence



Radical Change of our Society in its Full Breadth!

Impact on Mathematical Problem Settings

Some Examples:

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



Impact on Mathematical Problem Settings

Some Examples:

► Inverse Probleme/Imaging Science (2012–)

- ~ Denoising
- ~ Edge Detection
- ~ Inpainting
- ~ Classification
- ~ Superresolution
- ~ Limited-Angle Computed Tomography
- ~ ...



► Numerical Analysis of Partial Differential Equations (2017–)

- ~ Black-Scholes PDE
- ~ Allen-Cahn PDE
- ~ Parametric PDEs
- ~ ...



Artificial Intelligence = 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.



Problem with Reliability



Problems with Safety

Example:
Accidents involving robots



Problems with Security

Example:
Risks of hacking into AI systems



Problems with Privacy

Example:
Privacy violations of health data



Problems with Responsibility

Example:
Black-box and biased decisions

Problem with Reliability



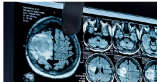
Problems with Safety

Example:
Accidents involving robots



Problems with Security

Example:
Risks of hacking into AI systems



Problems with Privacy

Example:
Privacy violations of health data



Problems with Responsibility

Example:
Black-box and biased decisions

Current major problem worldwide:

Lack of reliability of AI technology!

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 Artificial Intelligence!

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

Artificial Intelligence for Mathematics!

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



Delving Deeper into Artificial Intelligence...

First Appearance of Neural Networks

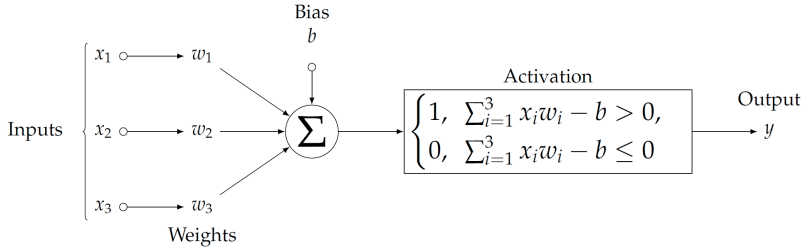
Key Task of McCulloch and Pitts (1943):

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

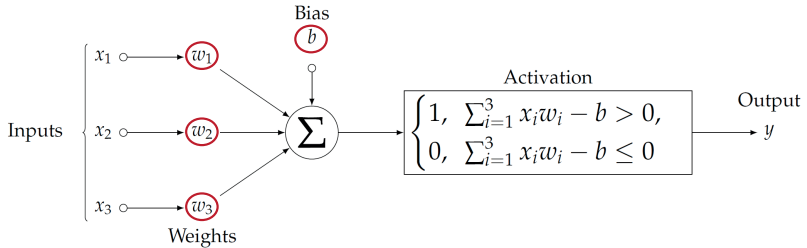
Goal: Artificial Intelligence!



Artificial Neurons



Artificial Neurons



Artificial Neurons

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

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

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

Artificial Neurons

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

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

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

Examples of Activation Functions:

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

Affine Linear Maps and Weights

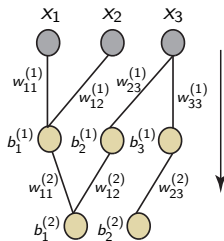
Remark: Concatenating artificial neurons leads to *compositions of affine linear maps and activation functions*.

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

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

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

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



Distinction: Architecture and Realization

Problem: Different architectures can lead to the same function!

Example: Let

$$W^{(1)} = \begin{pmatrix} \text{Id}_{\mathbb{R}^{N_0}} \\ -\text{Id}_{\mathbb{R}^{N_0}} \end{pmatrix}, \quad b^{(1)} = b^{(2)} = 0, \quad W^{(2)} = (\text{Id}_{\mathbb{R}^{N_0}}, -\text{Id}_{\mathbb{R}^{N_0}}).$$

Then, for all $x \in \mathbb{R}^{N_0}$,

$$\Phi(x) = W^{(2)} \text{ReLU}(W^{(1)}x + b^{(1)}) + b^{(2)} = \text{ReLU}(x) - \text{ReLU}(-x) = x.$$

Distinction: Architecture and Realization

Problem: Different architectures can lead to the same function!

Example: Let

$$W^{(1)} = \begin{pmatrix} \text{Id}_{\mathbb{R}^{N_0}} \\ -\text{Id}_{\mathbb{R}^{N_0}} \end{pmatrix}, \quad b^{(1)} = b^{(2)} = 0, \quad W^{(2)} = (\text{Id}_{\mathbb{R}^{N_0}}, -\text{Id}_{\mathbb{R}^{N_0}}).$$

Then, for all $x \in \mathbb{R}^{N_0}$,

$$\Phi(x) = W^{(2)} \text{ReLU}(W^{(1)}x + b^{(1)}) + b^{(2)} = \text{ReLU}(x) - \text{ReLU}(-x) = x.$$

*Solution: We distinguish
between architecture and realization of neural networks!*

Definition of a Deep Neural Network, Part 1

Definition:

A *fully connected feedforward neural network* is given by its *architecture*

$$a = (N, \varrho),$$

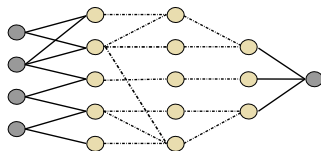
where $L \in \mathbb{N}$, $N \in \mathbb{N}^{L+1}$, and $\varrho: \mathbb{R} \rightarrow \mathbb{R}$.

We refer to

- ▶ ϱ as the *activation function*,
- ▶ L as the *number of layers*, and
- ▶ N_0 , N_L , and N_ℓ , $\ell \in [L-1]$, as the *number of neurons in the input, output, and ℓ -th hidden layer*, respectively.

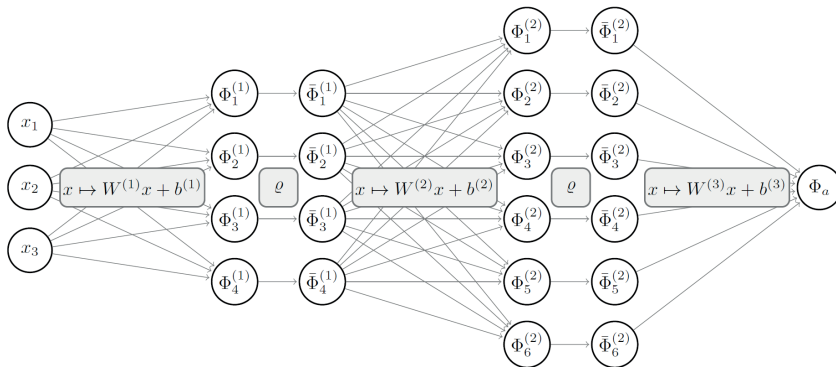
We denote the *number of parameters* by

$$P(N) := \sum_{\ell=1}^L N_\ell N_{\ell-1} + N_\ell$$



Definition of a Deep Neural Network, Illustration

Deep neural network with architecture $a = ((3, 4, 6, 1), \varrho)$:



Definition of a Deep Neural Network, Part 2

Definition (continued):

We define the corresponding *realization function* $\Phi_a: \mathbb{R}^{N_0} \times \mathbb{R}^{P(N)} \rightarrow \mathbb{R}^{N_L}$, which satisfies for every input $x \in \mathbb{R}^{N_0}$ and parameters

$$\theta = (\theta^{(\ell)})_{\ell=1}^L = ((W^{(\ell)}, b^{(\ell)}))_{\ell=1}^L \in \prod_{\ell=1}^L (\mathbb{R}^{N_\ell \times N_{\ell-1}} \times \mathbb{R}^{N_\ell}) \cong \mathbb{R}^{P(N)}$$

that $\Phi_a(x, \theta) = \Phi^{(L)}(x, \theta)$, where

$$\Phi^{(1)}(x, \theta) = W^{(1)}x + b^{(1)},$$

$$\bar{\Phi}^{(\ell)}(x, \theta) = \varrho(\Phi^{(\ell)}(x, \theta)), \quad \ell \in [L-1], \quad \text{and}$$

$$\Phi^{(\ell+1)}(x, \theta) = W^{(\ell+1)}\bar{\Phi}^{(\ell)}(x, \theta) + b^{(\ell+1)}, \quad \ell \in [L-1].$$

Definition of a Deep Neural Network, Part 2

Definition (continued):

We define the corresponding *realization function* $\Phi_a: \mathbb{R}^{N_0} \times \mathbb{R}^{P(N)} \rightarrow \mathbb{R}^{N_L}$, which satisfies for every input $x \in \mathbb{R}^{N_0}$ and parameters

$$\theta = (\theta^{(\ell)})_{\ell=1}^L = ((W^{(\ell)}, b^{(\ell)}))_{\ell=1}^L \in \prod_{\ell=1}^L (\mathbb{R}^{N_\ell \times N_{\ell-1}} \times \mathbb{R}^{N_\ell}) \cong \mathbb{R}^{P(N)}$$

that $\Phi_a(x, \theta) = \Phi^{(L)}(x, \theta)$, where

$$\Phi^{(1)}(x, \theta) = W^{(1)}x + b^{(1)},$$

$$\bar{\Phi}^{(\ell)}(x, \theta) = \varrho(\Phi^{(\ell)}(x, \theta)), \quad \ell \in [L-1], \quad \text{and}$$

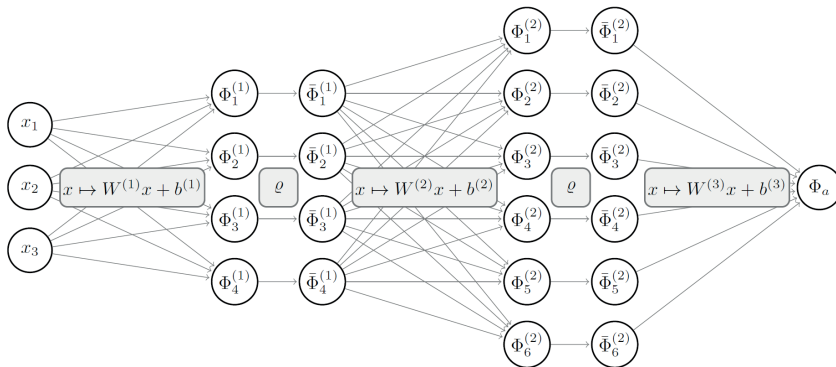
$$\Phi^{(\ell+1)}(x, \theta) = W^{(\ell+1)}\bar{\Phi}^{(\ell)}(x, \theta) + b^{(\ell+1)}, \quad \ell \in [L-1].$$

We refer to

- ▶ $W^{(\ell)} \in \mathbb{R}^{N_\ell \times N_{\ell-1}}$ and $b^{(\ell)} \in \mathbb{R}^{N_\ell}$ as the *weight matrices* and *bias vectors*, and to
- ▶ $\bar{\Phi}^{(\ell)}$ and $\Phi^{(\ell)}$ as the *activations* and *pre-activations* of the N_ℓ neurons in the ℓ -th layer.

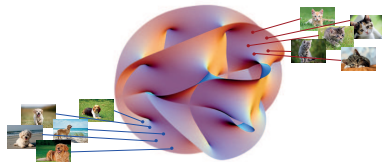
Definition of a Deep Neural Network, Illustration

Deep neural network with architecture $a = ((3, 4, 6, 1), \varrho)$:



High-Level Set Up:

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



Training of Deep Neural Networks

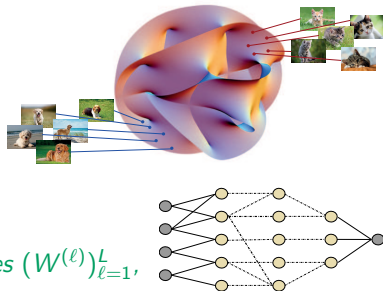
High-Level Set Up:

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

\leadsto Training- and test data set.

- Select an architecture $a = (N, \varrho)$ of a deep neural network.

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 $(x_i, f(x_i))_{i=1}^{\tilde{m}}$ of a function such as $f : \mathcal{M} \rightarrow \{1, 2, \dots, K\}$.

\leadsto Training- and test data set.

- ▶ Select an architecture $a = (N, \varrho)$ of a deep neural network.

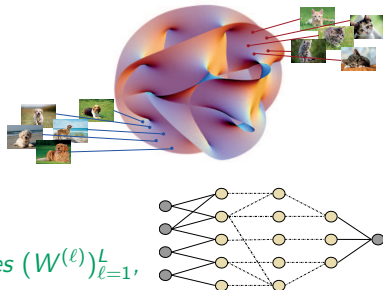
Sometimes selected entries of the matrices $(W^{(\ell)})_{\ell=1}^L$, i.e., weights, are set to zero at this point.

- ▶ Learn the weight matrices $W^{(\ell)} \in \mathbb{R}^{N_\ell \times N_{\ell-1}}$ and the bias vectors $b^{(\ell)} \in \mathbb{R}^{N_\ell}$ by

$$\min_{\theta} \sum_{i=1}^m \mathcal{L}(\Phi_a(x_i, \theta), f(x_i)) + \lambda \mathcal{P}(\theta)$$

yielding the network Φ_a .

This is often done by stochastic gradient descent.



Training of Deep Neural Networks

High-Level Set Up:

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

\leadsto Training- and test data set.

- ▶ Select an architecture $a = (N, \varrho)$ of a deep neural network.

Sometimes selected entries of the matrices $(W^{(\ell)})_{\ell=1}^L$, i.e., weights, are set to zero at this point.

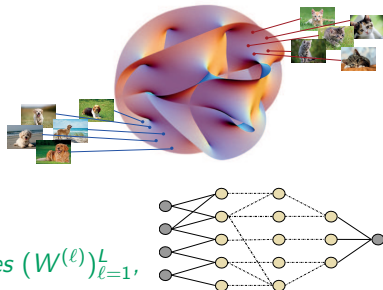
- ▶ Learn the weight matrices $W^{(\ell)} \in \mathbb{R}^{N_\ell \times N_{\ell-1}}$ and the bias vectors $b^{(\ell)} \in \mathbb{R}^{N_\ell}$ by

$$\min_{\theta} \sum_{i=1}^m \mathcal{L}(\Phi_a(x_i, \theta), f(x_i)) + \lambda \mathcal{P}(\theta)$$

yielding the network Φ_a .

This is often done by stochastic gradient descent.

Goal: $\Phi_a(\cdot, \theta) \approx f$



► Expressivity:

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

↪ *Applied Harmonic Analysis, Approximation Theory, ...*

► Expressivity:

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

↪ *Applied Harmonic Analysis, Approximation Theory, ...*

► Learning:

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

↪ *Algebraic/Differential Geometry, Optimal Control, Optimization, ...*

► Expressivity:

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

~> *Applied Harmonic Analysis, Approximation Theory, ...*

► Learning:

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

~> *Algebraic/Differential Geometry, Optimal Control, Optimization, ...*

► Generalization:

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

~> *Learning Theory, Probability Theory, Statistics, ...*

► Expressivity:

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

~ Applied Harmonic Analysis, Approximation Theory, ...

► Learning:

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

~ Algebraic/Differential Geometry, Optimal Control, Optimization, ...

► Generalization:

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

~ 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?

~ Information Theory, Uncertainty Quantification, ...

► Expressivity: (Lecture 1)

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

~ Applied Harmonic Analysis, Approximation Theory, ...

► Learning:

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

~ Algebraic/Differential Geometry, Optimal Control, Optimization, ...

► Generalization:

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

~ 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?

~ Information Theory, Uncertainty Quantification, ...

► 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?

↪ *Imaging Science, Inverse Problems, Microlocal Analysis, ...*

► 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?

~> *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?

~> *Numerical Mathematics, Partial Differential Equations, ...*

► 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?

~> *Imaging Science, Inverse Problems, Microlocal Analysis, ...*

► Partial Differential Equations: (Lecture 2)

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

~> *Numerical Mathematics, Partial Differential Equations, ...*

Lecture 2: Limitations and Next Generation AI

Expressivity

High-Dimensional Approximation in the AI World

The Approximation Error

Key Questions:

- ▶ What is the expressive power of a *given architecture*?
- ▶ What effect has the *depth* of a neural network in this respect?
- ▶ What is the *complexity* of the approximating neural network?
- ▶ What are *suitable function spaces* to consider?

The Approximation Error

Key Questions:

- ▶ What is the expressive power of a *given architecture*?
- ▶ What effect has the *depth* of a neural network in this respect?
- ▶ What is the *complexity* of the approximating neural network?
- ▶ What are *suitable function spaces* to consider?

Definition:

The *complexity* of a deep neural network with architecture $a = (N, \varrho)$, weight matrices $W^{(\ell)} \in \mathbb{R}^{N_\ell \times N_{\ell-1}}$, $1 \leq \ell \leq L$, and bias vectors $b^{(\ell)} \in \mathbb{R}^{N_\ell}$, $1 \leq \ell \leq L$, is defined by

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

Revisiting 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

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

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

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

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*

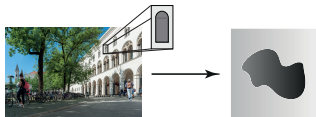
Modeling Anisotropic Structures

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 $\text{supp } f_i \subseteq [0, 1]^2$ and $\|f_i\|_{C^2} \leq 1$, $i = 0, 1$.



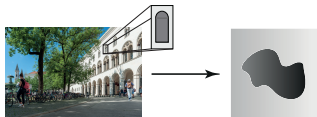
Modeling Anisotropic Structures

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 $\text{supp } f_i \subseteq [0, 1]^2$ 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 $f \in \mathcal{E}^2(\mathbb{R}^2)$:

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

Review of 2-D Wavelets

Definition (1D): Let $\phi \in L^2(\mathbb{R})$ be a scaling function and $\psi \in L^2(\mathbb{R})$ be a wavelet. Then the associated *wavelet system* is defined by

$$\{\phi(x - m) : m \in \mathbb{Z}\} \cup \{2^{j/2} \psi(2^j x - m) : j \geq 0, m \in \mathbb{Z}\}.$$



Review of 2-D Wavelets

Definition (1D): Let $\phi \in L^2(\mathbb{R})$ be a scaling function and $\psi \in L^2(\mathbb{R})$ be a wavelet. Then the associated **wavelet system** is defined by

$$\{\phi(x - m) : m \in \mathbb{Z}\} \cup \{2^{j/2} \psi(2^j x - m) : j \geq 0, m \in \mathbb{Z}\}.$$



Definition (2D): A **wavelet system** is defined by

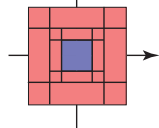
$$\{\phi^{(1)}(x - m) : m \in \mathbb{Z}^2\} \cup \{2^j \psi^{(i)}(2^j x - m) : j \geq 0, m \in \mathbb{Z}^2, i = 1, 2, 3\},$$

where

$$\psi^{(1)}(x) = \phi(x_1)\psi(x_2),$$

$$\phi^{(1)}(x) = \phi(x_1)\phi(x_2) \quad \text{and} \quad \psi^{(2)}(x) = \psi(x_1)\phi(x_2),$$

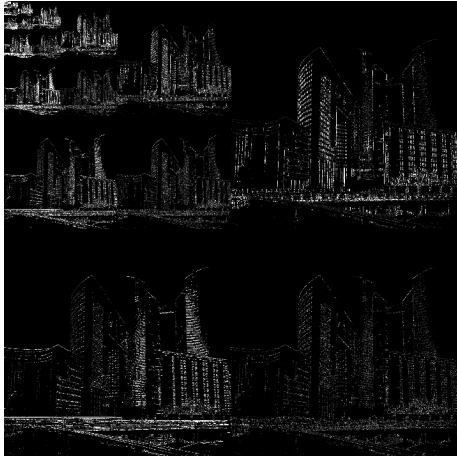
$$\psi^{(3)}(x) = \psi(x_1)\psi(x_2).$$



Theorem: Wavelets provide optimally sparse approximations for functions $f \in L^2(\mathbb{R}^2)$, which are C^2 apart from point singularities:

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

Wavelet Decomposition: JPEG2000



Wavelet Decomposition: JPEG2000



Original



25% Compression



5% Compression

What can Wavelets do?

Problem:

- *Isotropic* structure of wavelets:

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

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

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



What can Wavelets do?

Problem:

- *Isotropic* structure of wavelets:

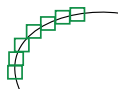
$$\{2^j \psi\left(\begin{pmatrix} 2^j & 0 \\ 0 & 2^j \end{pmatrix} x - m\right) : j \in \mathbb{Z}, m \in \mathbb{Z}^2\}, \quad \psi \in L^2(\mathbb{R}^2).$$

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

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

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)



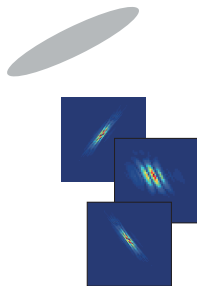
(Cone-adapted) Discrete Shearlet Systems

Parabolic scaling ('width \approx length²):

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

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 length²):

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

Orientation via shearing:

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

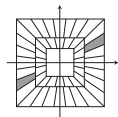
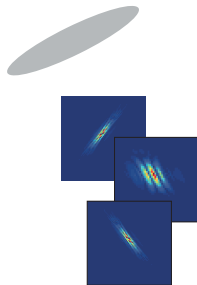
Definition (K, Labate; 2006):

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.

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.,

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



Optimally Sparse Approximation

Theorem (K, Lim; 2011):

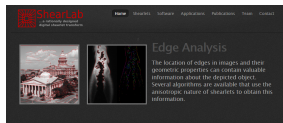
Let $\phi, \psi, \tilde{\psi} \in L^2(\mathbb{R}^2)$ be compactly supported, and let $\hat{\psi}, \hat{\tilde{\psi}}$ satisfy certain decay condition. Then $\mathcal{SH}(\phi, \psi, \tilde{\psi})$ provides an *optimally sparse approximation* of $f \in \mathcal{E}^2(\mathbb{R}^2)$, i.e.,

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



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

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



Welcome to shearlab.org

ShearLab is a MATLAB library developed for processing 2D and 3D data with a mixture of basis functions named shearlets. Such shearlet systems are particularly well adapted to represent **anisotropic features** (such as curves) that are often crucial in multidimensional data. The resulting representation has proven well-suited for **image processing** tasks such as inpainting, denoising or image segmentation. On this website we provide the full MATLAB code, a framework for numerical tests as well as general information on shearlets.

Similar to wavelet systems, shearlet systems are constructed by modifying generator functions. For wavelet systems, these functions

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

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

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

Universality of Shallow Neural Networks

Remark: Assume ϱ is a polynomial of degree q . Then $\varrho(Wx + b)$ is also a polynomial of degree q , hence Φ is also a polynomial of degree $\leq L \cdot q$. Hence in this case $C(\mathbb{R}^d)$ cannot be well approximated.

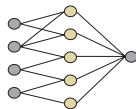
Universality of Shallow Neural Networks

Remark: Assume ϱ is a polynomial of degree q . Then $\varrho(Wx + b)$ is also a polynomial of degree q , hence Φ is also a polynomial of degree $\leq L \cdot q$. Hence in this case $C(\mathbb{R}^d)$ cannot be well approximated.

Universal Approximation Theorem (Cybenko, 1989)(Hornik, 1991):

Let $K \subset \mathbb{R}^d$ compact, $f : K \rightarrow \mathbb{R}$ continuous, $\varrho : \mathbb{R} \rightarrow \mathbb{R}$ continuous and not a polynomial. Then, for each $\epsilon > 0$, there exist $N \in \mathbb{N}$, $a_k, b_k \in \mathbb{R}, w_k \in \mathbb{R}^d$ with

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



Every continuous function on a compact set can be arbitrarily well approximated with a neural network with one single hidden layer.

Idea of Proof

- ▶ For $d \geq 1$, ϱ continuous, $\varrho : \mathbb{R} \rightarrow \mathbb{R}$ TFAE:
 - (i) $\text{span}\{\varrho(\langle w, x \rangle - b) : w \in \mathbb{R}^d, b \in \mathbb{R}\}$ is dense $C(K, \mathbb{R})$.
 - (ii) ϱ is not a polynomial.
- ▶ Now: (ii) \Rightarrow (i) for $d = 1$ and a smooth activation function ϱ .

Idea of Proof

- ▶ For $d \geq 1$, ϱ continuous, $\varrho : \mathbb{R} \rightarrow \mathbb{R}$ TFAE:
 - (i) $\text{span}\{\varrho(\langle w, x \rangle - b) : w \in \mathbb{R}^d, b \in \mathbb{R}\}$ is dense $C(K, \mathbb{R})$.
 - (ii) ϱ is not a polynomial.
- ▶ Now: (ii) \Rightarrow (i) for $d = 1$ and a smooth activation function ϱ .
- ▶ Since ϱ is not a polynomial, there exists one $x_0 \in \mathbb{R}$ with

$$\varrho^{(k)}(-x_0) \neq 0 \text{ for all } k.$$

Idea of Proof

- ▶ For $d \geq 1$, ϱ continuous, $\varrho : \mathbb{R} \rightarrow \mathbb{R}$ TFAE:
 - (i) $\text{span}\{\varrho(\langle w, x \rangle - b) : w \in \mathbb{R}^d, b \in \mathbb{R}\}$ is dense $C(K, \mathbb{R})$.
 - (ii) ϱ is not a polynomial.
- ▶ Now: (ii) \Rightarrow (i) for $d = 1$ and a smooth activation function ϱ .
- ▶ Since ϱ is not a polynomial, there exists one $x_0 \in \mathbb{R}$ with

$$\varrho^{(k)}(-x_0) \neq 0 \text{ for all } k.$$

- ▶ Constant functions can be arbitrarily well approximated:

$$\varrho(hx - x_0) \rightarrow \varrho(-x_0) \text{ as } h \rightarrow 0.$$

Idea of Proof

- ▶ For $d \geq 1$, ϱ continuous, $\varrho : \mathbb{R} \rightarrow \mathbb{R}$ TFAE:
 - (i) $\text{span}\{\varrho(\langle w, x \rangle - b) : w \in \mathbb{R}^d, b \in \mathbb{R}\}$ is dense $C(K, \mathbb{R})$.
 - (ii) ϱ is not a polynomial.
- ▶ Now: (ii) \Rightarrow (i) for $d = 1$ and a smooth activation function ϱ .
- ▶ Since ϱ is not a polynomial, there exists one $x_0 \in \mathbb{R}$ with

$$\varrho^{(k)}(-x_0) \neq 0 \text{ for all } k.$$

- ▶ Constant functions can be arbitrarily well approximated:

$$\varrho(hx - x_0) \rightarrow \varrho(-x_0) \text{ as } h \rightarrow 0.$$

- ▶ Linear functions can be arbitrarily well approximated:

$$\underbrace{\frac{\varrho((\lambda + h)x - x_0) - \varrho(x - x_0)}{h}}_{\rightarrow x \varrho'(\lambda x - x_0) \text{ for } h \rightarrow 0} \rightarrow x \cdot \varrho'(-x_0), \quad \text{as } h, \lambda \rightarrow 0.$$

Idea of Proof

- ▶ For $d \geq 1$, ϱ continuous, $\varrho : \mathbb{R} \rightarrow \mathbb{R}$ TFAE:
 - (i) $\text{span}\{\varrho(\langle w, x \rangle - b) : w \in \mathbb{R}^d, b \in \mathbb{R}\}$ is dense $C(K, \mathbb{R})$.
 - (ii) ϱ is not a polynomial.
- ▶ Now: (ii) \Rightarrow (i) for $d = 1$ and a smooth activation function ϱ .
- ▶ Since ϱ is not a polynomial, there exists one $x_0 \in \mathbb{R}$ with

$$\varrho^{(k)}(-x_0) \neq 0 \text{ for all } k.$$

- ▶ Constant functions can be arbitrarily well approximated:

$$\varrho(hx - x_0) \rightarrow \varrho(-x_0) \text{ as } h \rightarrow 0.$$

- ▶ Linear functions can be arbitrarily well approximated:

$$\underbrace{\frac{\varrho((\lambda + h)x - x_0) - \varrho(x - x_0)}{h}}_{\rightarrow x \varrho'(\lambda x - x_0) \text{ for } h \rightarrow 0} \rightarrow x \cdot \varrho'(-x_0), \quad \text{as } h, \lambda \rightarrow 0.$$

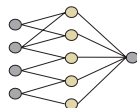
- \leadsto Any polynomial can be well approximated, then use Stone-Weierstraß
- \leadsto Finally, extend to d arbitrary.

Universality of Shallow Neural Networks

Universal Approximation Theorem (Cybenko, 1989)(Hornik, 1991):

Let $K \subset \mathbb{R}^d$ compact, $f : K \rightarrow \mathbb{R}$ continuous, $\varrho : \mathbb{R} \rightarrow \mathbb{R}$ continuous and not a polynomial. Then, for each $\epsilon > 0$, there exist $N \in \mathbb{N}$, $a_k, b_k \in \mathbb{R}, w_k \in \mathbb{R}^d$ with

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

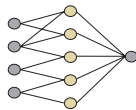


Universality of Shallow Neural Networks

Universal Approximation Theorem (Cybenko, 1989)(Hornik, 1991):

Let $K \subset \mathbb{R}^d$ compact, $f : K \rightarrow \mathbb{R}$ continuous, $\varrho : \mathbb{R} \rightarrow \mathbb{R}$ continuous and not a polynomial. Then, for each $\epsilon > 0$, there exist $N \in \mathbb{N}$, $a_k, b_k \in \mathbb{R}$, $w_k \in \mathbb{R}^d$ with

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



Corollary:

In this situation, we obtain

$$\epsilon^{\text{approx}} \rightarrow 0$$

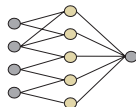
for increasing complexity of the neural networks.

Universality of Shallow Neural Networks

Universal Approximation Theorem (Cybenko, 1989)(Hornik, 1991):

Let $K \subset \mathbb{R}^d$ compact, $f : K \rightarrow \mathbb{R}$ continuous, $\varrho : \mathbb{R} \rightarrow \mathbb{R}$ continuous and not a polynomial. Then, for each $\epsilon > 0$, there exist $N \in \mathbb{N}$, $a_k, b_k \in \mathbb{R}$, $w_k \in \mathbb{R}^d$ with

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



Corollary:

In this situation, we obtain

$$\epsilon^{\text{approx}} \rightarrow 0$$

for increasing complexity of the neural networks.

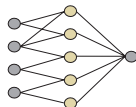
Approximation accuracy \leftrightarrow Complexity of approximating network?

Universality of Shallow Neural Networks

Universal Approximation Theorem (Cybenko, 1989)(Hornik, 1991):

Let $K \subset \mathbb{R}^d$ compact, $f : K \rightarrow \mathbb{R}$ continuous, $\varrho : \mathbb{R} \rightarrow \mathbb{R}$ continuous and not a polynomial. Then, for each $\epsilon > 0$, there exist $N \in \mathbb{N}$, $a_k, b_k \in \mathbb{R}, w_k \in \mathbb{R}^d$ with

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



Corollary:

In this situation, we obtain

$$\epsilon^{\text{approx}} \rightarrow 0$$

for increasing complexity of the neural networks.

Approximation accuracy \leftrightarrow Complexity of approximating network?

What about even optimality?

Lower Bounds for Approximation

Classical Approach:

- ▶ VC Dimension

Classical Approach:

- ▶ VC Dimension

Towards Optimal Complexity:

- ▶ How well can functions be approximated by neural networks with few non-zero weights?
 - ▶ Can we derive a lower bound on the necessary number of weights?
 - ▶ Can we construct neural networks which attain this bound?
- ▶ Are neural networks as good approximators as wavelets and shearlets?

Measure for Complexity of Function Class

Intuitive Definition:

The *optimal exponent* $\gamma^*(\mathcal{C})$ is a measure of complexity of the function class \mathcal{C} :

“The optimal exponent describes the dependence of the code length for encoding the function class on the required approximation quality.”

Theorem:

For $\mathcal{C} \subseteq L^2(\mathbb{R}^d)$, the optimal N –term approximation rate is given by

$$N^{-\frac{1}{\gamma^*(\mathcal{C})}}.$$

Definition:

- Let $d \in \mathbb{N}$, $\Omega \in \mathbb{R}^d$ and $\mathcal{C} \subset L^2(\Omega)$. For any $l \in \mathbb{N}$

$$\mathcal{E}^l = \{E : \mathcal{C} \rightarrow \{0, 1\}^l\}$$

is called the set of *binary encoders of length l* and

$$\mathcal{D}^l = \{D : \{0, 1\}^l \rightarrow L^2(\Omega)\}$$

is called the set of *binary decoders of length l* .

Definition:

- ▶ Let $d \in \mathbb{N}$, $\Omega \in \mathbb{R}^d$ and $\mathcal{C} \subset L^2(\Omega)$. For any $l \in \mathbb{N}$

$$\mathcal{E}^l = \{E : \mathcal{C} \rightarrow \{0, 1\}^l\}$$

is called the set of *binary encoders of length l* and

$$\mathcal{D}^l = \{D : \{0, 1\}^l \rightarrow L^2(\Omega)\}$$

is called the set of *binary decoders of length l* .

- ▶ A pair $(E, D) \in \mathcal{E}^l \times \mathcal{D}^l$ achieves *distortion $\varepsilon > 0$ over \mathcal{C}* , if

$$\sup_{f \in \mathcal{C}} \|D(E(f)) - f\|_{L^2} \leq \varepsilon.$$

Rate Distortion Theory

Definition:

- ▶ Let $d \in \mathbb{N}$, $\Omega \in \mathbb{R}^d$ and $\mathcal{C} \subset L^2(\Omega)$. For any $l \in \mathbb{N}$

$$\mathcal{E}^l = \{E : \mathcal{C} \rightarrow \{0, 1\}^l\}$$

is called the set of *binary encoders of length l* and

$$\mathcal{D}^l = \{D : \{0, 1\}^l \rightarrow L^2(\Omega)\}$$

is called the set of *binary decoders of length l* .

- ▶ A pair $(E, D) \in \mathcal{E}^l \times \mathcal{D}^l$ achieves *distortion $\varepsilon > 0$ over \mathcal{C}* , if

$$\sup_{f \in \mathcal{C}} \|D(E(f)) - f\|_{L^2} \leq \varepsilon.$$

- ▶ For $\varepsilon > 0$, the *minimal code length $L(\varepsilon, \mathcal{C})$* is

$$L(\varepsilon, \mathcal{C}) = \min\{l \in \mathbb{N} : \exists (E, D) \in \mathcal{E}^l \times \mathcal{D}^l : \sup_{f \in \mathcal{C}} \|D(E(f)) - f\|_{L^2} \leq \varepsilon\}.$$

The *optimal exponent $\gamma^*(\mathcal{C})$* is

$$\gamma^*(\mathcal{C}) := \inf\{\gamma \in \mathbb{R} : L(\varepsilon, \mathcal{C}) = O(\varepsilon^{-\gamma})\}.$$

A Fundamental Lower Bound

Theorem (Bölcskei, Grohs, K, and Petersen; 2017):

Let $d \in \mathbb{N}$, $\varrho : \mathbb{R} \rightarrow \mathbb{R}$, and let $\mathcal{C} \subset L^2(\mathbb{R}^d)$. Assume that

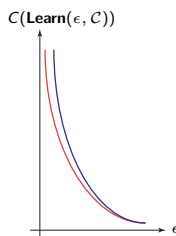
$$\mathbf{Learn} : (0, 1) \times \mathcal{C} \rightarrow \mathcal{F}_a$$

satisfies

$$\sup_{f \in \mathcal{C}} \|f - \mathbf{Learn}(\epsilon, f)\|_{L^2} \leq \epsilon.$$

Then, for all $\gamma < \gamma^*(\mathcal{C})$, there is no $C > 0$ with

$$\sup_{f \in \mathcal{C}} C(\mathbf{Learn}(\epsilon, f)) \leq C \epsilon^{-\gamma} \quad \text{for all } \epsilon > 0.$$



A Fundamental Lower Bound

Theorem (Bölcskei, Grohs, K, and Petersen; 2017):

Let $d \in \mathbb{N}$, $\varrho : \mathbb{R} \rightarrow \mathbb{R}$, and let $\mathcal{C} \subset L^2(\mathbb{R}^d)$. Assume that

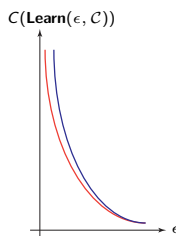
$$\mathbf{Learn} : (0, 1) \times \mathcal{C} \rightarrow \mathcal{F}_a$$

satisfies

$$\sup_{f \in \mathcal{C}} \|f - \mathbf{Learn}(\epsilon, f)\|_{L^2} \leq \epsilon.$$

Then, for all $\gamma < \gamma^*(\mathcal{C})$, there is no $C > 0$ with

$$\sup_{f \in \mathcal{C}} C(\mathbf{Learn}(\epsilon, f)) \leq C \epsilon^{-\gamma} \quad \text{for all } \epsilon > 0.$$



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

Optimally Sparse Deep Neural Networks

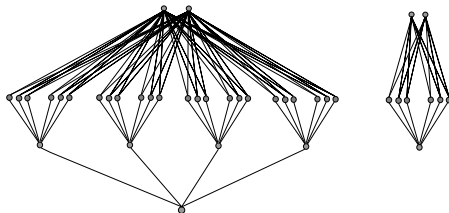
DNNs and Representation Systems, I

Observation: Assume a system $(\varphi_i)_{i \in I} \subset L^2(\mathbb{R}^d)$ satisfies:

- For each $i \in I$, there exists a neural network Φ_i with at most $C > 0$ edges such that $\varphi_i = \Phi_i$.

Then we can construct a network Φ with $O(M)$ edges with

$$\Phi = \sum_{i \in I_M} c_i \varphi_i, \quad \text{if } |I_M| = M.$$



Corollary: Assume a system $(\varphi_i)_{i \in I} \subset L^2(\mathbb{R}^d)$ satisfies:

- ▶ For each $i \in I$, there exists a neural network Φ_i with at most $C > 0$ edges such that $\varphi_i = \Phi_i$.
- ▶ There exists $\tilde{C} > 0$ such that, for all $f \in \mathcal{C} \subset L^2(\mathbb{R}^d)$, there exists $I_M \subset I$ with

$$\|f - \sum_{i \in I_M} c_i \varphi_i\| \leq \tilde{C} M^{-1/\gamma^*(\mathcal{C})}.$$

Then every $f \in \mathcal{C}$ can be approximated up to an error of ϵ by a neural network with only $O(\epsilon^{-\gamma^*(\mathcal{C})})$ edges.

Sketch of Proof:

- ▶ There exists a network Φ with $O(M)$ edges with $\Phi = \sum_{i \in I_M} c_i \varphi_i$.
- ▶ Set $\epsilon = \tilde{C} M^{-1/\gamma^*(\mathcal{C})}$ and solve for the number of edges M , yielding

$$M = O(\epsilon^{-\gamma^*(\mathcal{C})}).$$

Corollary: Assume a system $(\varphi_i)_{i \in I} \subset L^2(\mathbb{R}^d)$ satisfies:

- ▶ For each $i \in I$, there exists a neural network Φ_i with at most $C > 0$ edges such that $\varphi_i = \Phi_i$.
- ▶ There exists $\tilde{C} > 0$ such that, for all $f \in \mathcal{C} \subset L^2(\mathbb{R}^d)$, there exists $I_M \subset I$ with

$$\|f - \sum_{i \in I_M} c_i \varphi_i\| \leq \tilde{C} M^{-1/\gamma^*(\mathcal{C})}.$$

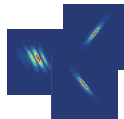
Then every $f \in \mathcal{C}$ can be approximated up to an error of ϵ by a neural network with only $O(\epsilon^{-\gamma^*(\mathcal{C})})$ edges.

Recall: If a neural network stems from a fixed learning procedure **Learn**, then, for all $\gamma < \gamma^*(\mathcal{C})$, there does not exist $C > 0$ such that

$$\sup_{f \in \mathcal{C}} C(\mathbf{Learn}(\epsilon, f)) \leq C \epsilon^{-\gamma} \quad \text{for all } \epsilon > 0.$$

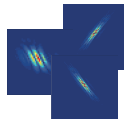
General Approach:

- (1) Determine a class of functions $\mathcal{C} \subseteq L^2(\mathbb{R}^2)$.
- (2) Determine an associated representation system with the following properties:
 - ▶ The elements of this system can be realized by a neural network with controlled number of edges.
 - ▶ This system provides optimally sparse approximations for \mathcal{C} .



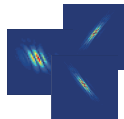
General Approach:

- (1) Determine a class of functions $\mathcal{C} \subseteq L^2(\mathbb{R}^2)$.
 \leadsto *Cartoon-like functions!*
- (2) Determine an associated representation system with the following properties:
 - ▶ The elements of this system can be realized by a neural network with controlled number of edges.
 - ▶ This system provides optimally sparse approximations for \mathcal{C} .



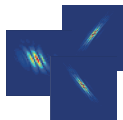
General Approach:

- (1) Determine a class of functions $\mathcal{C} \subseteq L^2(\mathbb{R}^2)$.
 \leadsto *Cartoon-like functions!*
- (2) Determine an associated representation system with the following properties:
 \leadsto *Shearlets!*
 - ▶ The elements of this system can be realized by a neural network with controlled number of edges.
 - ▶ This system provides optimally sparse approximations for \mathcal{C} .



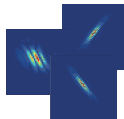
General Approach:

- (1) Determine a class of functions $\mathcal{C} \subseteq L^2(\mathbb{R}^2)$.
 \leadsto *Cartoon-like functions!*
- (2) Determine an associated representation system with the following properties:
 \leadsto *Shearlets!*
 - ▶ The elements of this system can be realized by a neural network with controlled number of edges.
 - ▶ This system provides optimally sparse approximations for \mathcal{C} .
 \leadsto *This has been proven!*



General Approach:

- (1) Determine a class of functions $\mathcal{C} \subseteq L^2(\mathbb{R}^2)$.
 \leadsto *Cartoon-like functions!*
- (2) Determine an associated representation system with the following properties:
 \leadsto *Shearlets!*
 - ▶ The elements of this system can be realized by a neural network with controlled number of edges.
 \leadsto *Still to be analyzed!*
 - ▶ This system provides optimally sparse approximations for \mathcal{C} .
 \leadsto *This has been proven!*



Construction of Generators

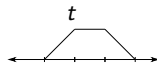
Wavelet generators (LeCun; 1987), (Shaham, Cloninger, Coifman; '17):

- ▶ Assume activation function $\varrho(x) = \max\{x, 0\}$ (ReLU).

- ▶ Define

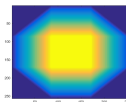
$$t(x) := \varrho(x) - \varrho(x-1) - \varrho(x-2) + \varrho(x-3).$$

$\leadsto t$ can be constructed with a 2 layer network.



- ▶ Observe that

$$\phi(x_1, x_2) := \varrho(t(x_1) + t(x_2) - 1)$$



yields a 2D bump function.

- ▶ Summing up shifted versions of ϕ yields a function ψ with vanishing moments.

$\leadsto \psi$ can be realized by a 3 layer neural network.

Construction of Generators

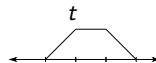
Wavelet generators (LeCun; 1987), (Shaham, Cloninger, Coifman; '17):

- ▶ Assume activation function $\varrho(x) = \max\{x, 0\}$ (ReLU).

- ▶ Define

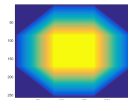
$$t(x) := \varrho(x) - \varrho(x-1) - \varrho(x-2) + \varrho(x-3).$$

$\leadsto t$ can be constructed with a 2 layer network.



- ▶ Observe that

$$\phi(x_1, x_2) := \varrho(t(x_1) + t(x_2) - 1)$$



yields a 2D bump function.

- ▶ Summing up shifted versions of ϕ yields a function ψ with vanishing moments.

$\leadsto \psi$ can be realized by a 3 layer neural network.

This cannot yield differentiable functions ψ !

Construction of Generators

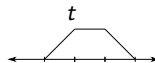
Wavelet generators (LeCun; 1987), (Shaham, Cloninger, Coifman; '17):

- ▶ Assume activation function $\varrho(x) = \max\{x, 0\}$ (ReLU).

- ▶ Define

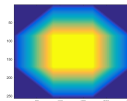
$$t(x) := \varrho(x) - \varrho(x-1) - \varrho(x-2) + \varrho(x-3).$$

~ t can be constructed with a 2 layer network.



- ▶ Observe that

$$\phi(x_1, x_2) := \varrho(t(x_1) + t(x_2) - 1)$$



yields a 2D bump function.

- ▶ Summing up shifted versions of ϕ yields a function ψ with vanishing moments.

~ psi can be realized by a 3 layer neural network.

Our Construction: Use a smoothed version of a ReLU.

~ Leads to appropriate shearlet generators!

Theorem (Bölcskei, Grohs, K, and Petersen; 2017): Let ϱ be an admissible smooth activation function, and let $\epsilon > 0$. Then there exist $C_\epsilon > 0$ such that, for all $f \in \mathcal{E}^2(\mathbb{R}^2)$ and $N \in \mathbb{N}$, we can construct a neural network Φ with 4 layers and complexity $C(\Phi) = O(N)$ satisfying

$$\|f - \Phi\|_{L^2(\mathbb{R}^2)} \leq C_\epsilon N^{-1+\epsilon}.$$

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

Theorem (Bölcskei, Grohs, K, and Petersen; 2017): Let ϱ be an admissible smooth activation function, and let $\epsilon > 0$. Then there exist $C_\epsilon > 0$ such that, for all $f \in \mathcal{E}^2(\mathbb{R}^2)$ and $N \in \mathbb{N}$, we can construct a neural network Φ with 4 layers and complexity $C(\Phi) = O(N)$ satisfying

$$\|f - \Phi\|_{L^2(\mathbb{R}^2)} \leq C_\epsilon N^{-1+\epsilon}.$$

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

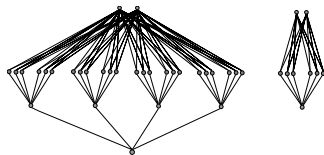
*Function classes which are optimal representable by shearlets, etc.
are also optimally approximated
by memory-efficient neural networks with a parallel architecture!*

Typically weights are learnt by backpropagation. This raises the following question:

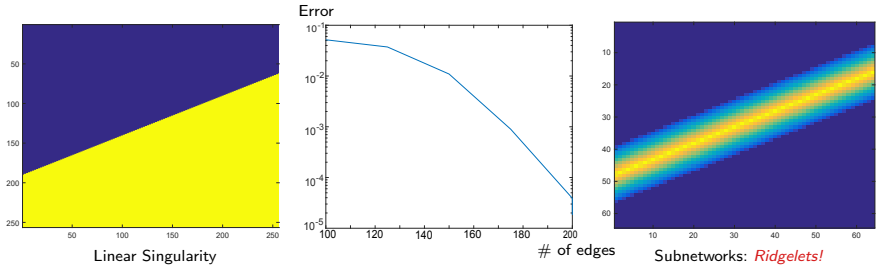
Does this lead to the optimal complexity?

Our setup:

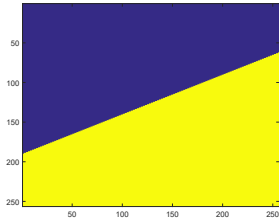
- ▶ Fixed network topology with ReLUs.
- ▶ Specific functions to learn.
- ▶ Learning through SGD.
- ▶ Analyze the learnt subnetworks.
- ▶ Analysis of the connection between approximation error and number of edges.



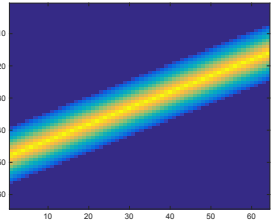
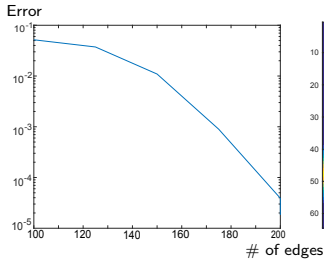
Numerical Experiments (with ReLUs & Backpropagation)



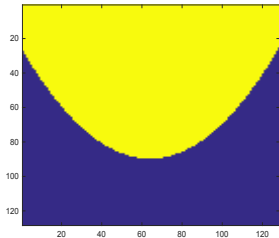
Numerical Experiments (with ReLUs & Backpropagation)



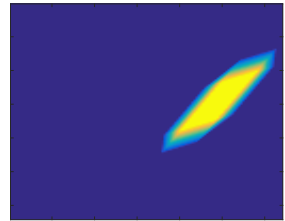
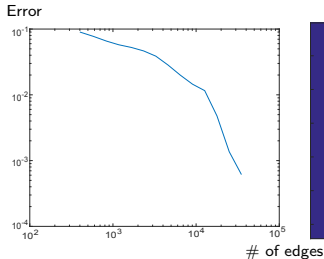
Linear Singularity



Subnetworks: *Ridgelets!*



Curvilinear Singularity



Subnetworks: \approx *Shearlets!*

The Role of Depth

Our Situation:

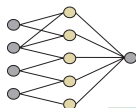
We now consider deep neural networks with *ReLU activation function* $\varrho_R(x) = \max\{0, x\}$.

Properties of ReLU Neural Networks:

- (1) A two-layer ReLU neural network with one-dimensional input and output is a function of the form

$$\Phi(x) := \sum_{i=1}^n w_i^{(2)} \varrho_R(w_i^{(1)} x + b_i^{(1)}) + b^{(2)}, \quad x \in \mathbb{R},$$

- where $w_i^{(1)}, w_i^{(2)}, b_i^{(1)}, b^{(2)} \in \mathbb{R}$ for $i \in [n]$.
- (2) Φ is a continuous piecewise affine linear function.

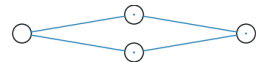
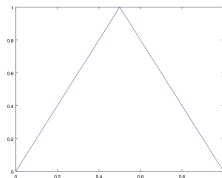


The Hat Function

General Observation:

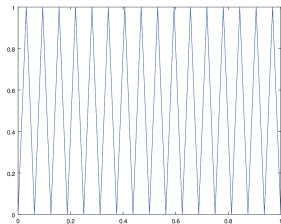
We can write the *hat function* $h: [0, 1] \rightarrow [0, 1]$ as a neural network with 2 layers and 2 neurons:

$$h(x) = 2\varrho_R(x) - 4\varrho_R(x - \tfrac{1}{2}) = \begin{cases} 2x, & \text{if } 0 \leq x < \tfrac{1}{2}, \\ 2(1 - x), & \text{if } \tfrac{1}{2} \leq x \leq 1. \end{cases}$$



Observation by (Telegarsky; 2016):

The n -fold convolution $h_n(x) := h \circ \dots \circ h$ produces a sawtooth function with 2^n spikes. In particular, h_n admits 2^n affine linear pieces with only $2n$ many neurons.



*Deep ReLU neural networks are exponentially more efficient
in generating affine linear pieces!*

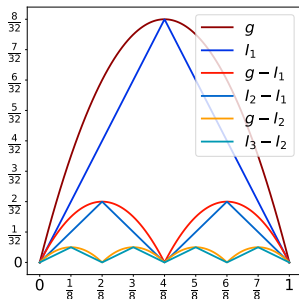
Approximating Smooth Functions

Idea:

- ▶ Let I_n be interpolation of $[0, 1] \ni x \mapsto g(x) := x - x^2$ on $2^n + 1$ equidistant points.
- ▶ I_n is a sum of n sawtooth functions:

$$I_n = \sum_{k=1}^n I_k - I_{k-1} = \sum_{k=1}^n \frac{h_k}{2^{2k}}.$$

- ▶ Each $h_k = h_{k-1} \circ h$ can be written as a k -fold composition of h .



Approximating Smooth Functions (Continued)

This leads to efficient approximation by ReLU neural networks of

- ▶ $x \mapsto x^2$,
- ▶ $(x, y) \mapsto xy$,
- ▶ localized Taylor polynomials,
- ▶ smooth functions,
- ▶ ...Sobolev-regular functions.

Theorem (Gühring, K, Petersen; 2020):

Let $d, k \in \mathbb{N}$ with $k \geq 2$, let $p \in [1, \infty]$, $s \in [0, 1]$, and $B \in (0, \infty)$. Then there exists a constant $c \in (0, \infty)$ with the following property: For every $\varepsilon \in (0, \frac{1}{2})$ there exists a neural network architecture $a = (N, \varrho)$ with

$$P(N) \leq c\varepsilon^{-d/(k-s)} \log(1/\varepsilon)$$

such that for every function $g \in W^{k,p}((0,1)^d)$ with $\|g\|_{W^{k,p}((0,1)^d)} \leq B$ it holds that

$$\inf_{\theta \in \mathbb{R}^{P(N)}} \|\Phi_a(\theta, \cdot) - g\|_{W^{s,p}((0,1)^d)} \leq \varepsilon.$$

Depth-Width Approximation Trade-Off

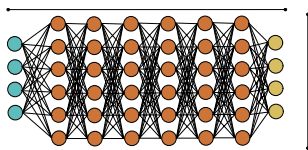
Theorem (Yarotzky; 2017):

Let $d, L \in \mathbb{N}$ with $L \geq 2$ and let $g \in C^2([0, 1]^d)$ be a function which is not affine linear. Then there exists a constant $c \in (0, \infty)$ with the following property: For every $\varepsilon \in (0, 1)$ and every ReLU neural network architecture $a = (N, \varrho_R) = ((d, N_1, \dots, N_{L-1}, 1), \varrho_R)$ with L layers and

$$\|N\|_1 \leq c\varepsilon^{-1/(2(L-1))}$$

neurons, then

$$\inf_{\theta \in \mathbb{R}^{P(N)}} \|\Phi_a(\cdot, \theta) - g\|_{L^\infty([0,1]^d)} \geq \varepsilon.$$



Depth-Width Approximation Trade-Off

Theorem (Yarotzky; 2017):

Let $d, L \in \mathbb{N}$ with $L \geq 2$ and let $g \in C^2([0, 1]^d)$ be a function which is not affine linear. Then there exists a constant $c \in (0, \infty)$ with the following property: For every $\varepsilon \in (0, 1)$ and every ReLU neural network architecture $a = (N, \varrho_R) = ((d, N_1, \dots, N_{L-1}, 1), \varrho_R)$ with L layers and

$$\|N\|_1 \leq c\varepsilon^{-1/(2(L-1))}$$

neurons, then

$$\inf_{\theta \in \mathbb{R}^{P(N)}} \|\Phi_a(\cdot, \theta) - g\|_{L^\infty([0,1]^d)} \geq \varepsilon.$$

Idea of Proof:

- ▶ Use the fact that ReLU neural networks are piecewise affine linear.
- ▶ Show that the number of pieces that can be generated using an architecture $((1, N_1, \dots, N_{L-1}, 1), \varrho_R)$ scales roughly like $\prod_{\ell=1}^{L-1} N_\ell$.

Alternative Notions of Expressivity

A Different Viewpoint

Instead of the classical approximation framework, alternative notions aim to relate *structural properties of the neural network* with the *richness of the set of possibly expressed functions*.

A Different Viewpoint

Instead of the classical approximation framework, alternative notions aim to relate *structural properties of the neural network* with the *richness of the set of possibly expressed functions*.

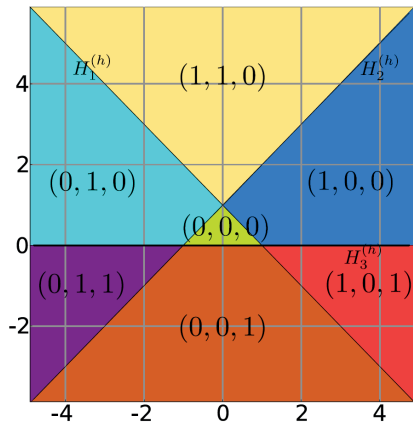
Early Work (Montúfar, Pascanu, Cho, Bengio; 2014):

- ▶ Consider *affine linear regions* of a ReLU neural network $\Phi_{(N, \varrho_R)}(\cdot, \theta)$.
- ▶ Analyze the growth of their number depending on the depth.

Definition:

Affine linear regions are the connected components of $\mathbb{R}^{N_0} \setminus H$, where H is the set of non-differentiability of the realization $\Phi_{(N, \varrho_R)}(\cdot, \theta)$.

Illustration of Affine-Linear Regions



Source: Hinz, van de Geer. A Framework for the construction of upper bounds on the number of affine linear regions of ReLU feed-forward neural networks. IEEE Transactions on Information Theory 65 (2019), 7304–7324

Growth of Number of Affine Linear Regions

Informal Theorem (Hinz, Sara van de Geer; 2019):

“Deep ReLU neural networks can exhibit significantly more regions than their shallow counterparts.”

Growth of Number of Affine Linear Regions

Informal Theorem (Hinz, Sara van de Geer; 2019):

“Deep ReLU neural networks can exhibit significantly more regions than their shallow counterparts.”

Intuition of the Effect of Depth:

- ▶ Through the ReLU each neuron $\mathbb{R}^d \ni x \mapsto \varrho_R(\langle x, w \rangle + b)$, $w \in \mathbb{R}^d$, $b \in \mathbb{R}$, splits the space into two affine linear regions separated by the hyperplane

$$\{x \in \mathbb{R}^d : \langle x, w \rangle + b = 0\}.$$

- ▶ A shallow ReLU neural network $\Phi_{((d,n,1),\varrho_R)}(\cdot, \theta)$ with n neurons in the hidden layer therefore produces a number of regions defined through n hyperplanes.
- ▶ One can bound the number of affine linear regions by $\sum_{j=0}^d \binom{n}{j}$.
- ▶ Deepening neural networks then corresponds to a folding of the input space.

Growth of Number of Affine Linear Regions

Informal Theorem (Hinz, Sara van de Geer; 2019):

“Deep ReLU neural networks can exhibit significantly more regions than their shallow counterparts.”

Intuition of the Effect of Depth:

- Through the ReLU each neuron $\mathbb{R}^d \ni x \mapsto \varrho_R(\langle x, w \rangle + b)$, $w \in \mathbb{R}^d$, $b \in \mathbb{R}$, splits the space into two affine linear regions separated by the hyperplane

$$\{x \in \mathbb{R}^d : \langle x, w \rangle + b = 0\}.$$

- A shallow ReLU neural network $\Phi_{((d,n,1),\varrho_R)}(\cdot, \theta)$ with n neurons in the hidden layer therefore produces a number of regions defined through n hyperplanes.
- One can bound the number of affine linear regions by $\sum_{j=0}^d \binom{n}{j}$.
- Deepening neural networks then corresponds to a folding of the input space.

This leads to an exponential efficiency of deep neural networks in generating affine linear regions!

Going one Step Further

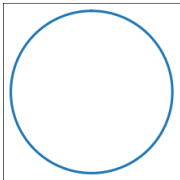
Idea:

How does the length of a non-constant curve in the input space changes in expectation through the layers of a neural network?

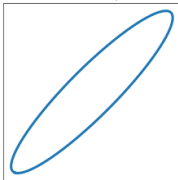
Illustration

Shape of the trajectory $t \mapsto \Phi_{((2,n,\dots,n,2),\varrho_R)}(\gamma(t), \theta)$ of the output of a randomly initialized network with 0, 3, 10 hidden layers:

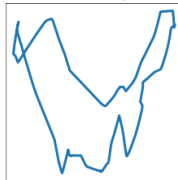
Input Curve



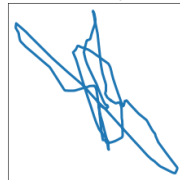
No hidden layer



Three hidden layers



Ten hidden layers



Going one Step Further

Idea:

How does the length of a non-constant curve in the input space changes in expectation through the layers of a neural network?

Going one Step Further

Idea:

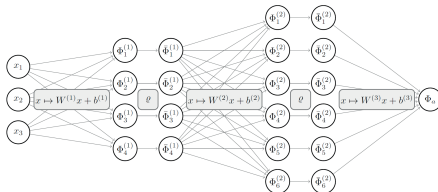
How does the length of a non-constant curve in the input space changes in expectation through the layers of a neural network?

Definition (Raghu, Poole, Kleinberg, Ganguli, Sohl-Dickstein; 2017):

Consider a ReLU network with architecture $a = ((N_0, n, \dots, n, N_L), \varrho_R)$ and depth $L \in \mathbb{N}$. Given a non-constant continuous curve $\gamma: [0, 1] \rightarrow \mathbb{R}^{N_0}$ in the input space, the *length of the trajectory* in the ℓ -th layer of the neural network $\Phi_a(\cdot, \theta)$ is then given by

$$\text{Length}(\bar{\Phi}^{(\ell)}(\gamma(\cdot), \theta)), \quad \ell \in [L - 1],$$

where $\bar{\Phi}^{(\ell)}(\cdot, \theta)$ is the activation in the ℓ -th layer.



Analyzing the Length of the Trajectory

Coarse Argumentation:

- ▶ Let the parameters Θ_1 of Φ_a be initialized independently
 - ▶ The entries corresponding to the weight matrices and bias vectors follow a normal distribution with zero mean and variances $1/n$ and 1 , respectively.
- ▶ We can then conclude that

$$\mathbb{E}[\text{Length}(\bar{\Phi}^{(\ell)}(\gamma(\cdot), \Theta_1))] = c > 0.$$

- ▶ Let $\sigma \in (0, \infty)$.
- ▶ Consider a second initialization Θ_σ , now with σ^2/n and σ^2 .
- ▶ By positive homogeneity of the ReLU, we obtain

$$\bar{\Phi}^{(\ell)}(\cdot, \Theta_\sigma) \sim \sigma^\ell \bar{\Phi}^{(\ell)}(\cdot, \Theta_1).$$

Analyzing the Length of the Trajectory

Coarse Argumentation:

- ▶ Let the parameters Θ_1 of Φ_a be initialized independently
 - ▶ The entries corresponding to the weight matrices and bias vectors follow a normal distribution with zero mean and variances $1/n$ and 1 , respectively.
- ▶ We can then conclude that

$$\mathbb{E}[\text{Length}(\bar{\Phi}^{(\ell)}(\gamma(\cdot), \Theta_1))] = c > 0.$$

- ▶ Let $\sigma \in (0, \infty)$.
- ▶ Consider a second initialization Θ_σ , now with σ^2/n and σ^2 .
- ▶ By positive homogeneity of the ReLU, we obtain

$$\bar{\Phi}^{(\ell)}(\cdot, \Theta_\sigma) \sim \sigma^\ell \bar{\Phi}^{(\ell)}(\cdot, \Theta_1).$$

Theorem (Raghu, Poole, Kleinberg, Ganguli, Sohl-Dickstein; 2017):

We have

$$\mathbb{E}[\text{Length}(\bar{\Phi}^{(\ell)}(\gamma(\cdot), \Theta_\sigma))] = \sigma^\ell c.$$

Analyzing the Length of the Trajectory

Coarse Argumentation:

- ▶ Let the parameters Θ_1 of Φ_a be initialized independently
 - ▶ The entries corresponding to the weight matrices and bias vectors follow a normal distribution with zero mean and variances $1/n$ and 1 , respectively.
- ▶ We can then conclude that

$$\mathbb{E}[\text{Length}(\bar{\Phi}^{(\ell)}(\gamma(\cdot), \Theta_1))] = c > 0.$$

- ▶ Let $\sigma \in (0, \infty)$.
- ▶ Consider a second initialization Θ_σ , now with σ^2/n and σ^2 .
- ▶ By positive homogeneity of the ReLU, we obtain

$$\bar{\Phi}^{(\ell)}(\cdot, \Theta_\sigma) \sim \sigma^\ell \bar{\Phi}^{(\ell)}(\cdot, \Theta_1).$$

Theorem (Raghu, Poole, Kleinberg, Ganguli, Sohl-Dickstein; 2017):

We have

$$\mathbb{E}[\text{Length}(\bar{\Phi}^{(\ell)}(\gamma(\cdot), \Theta_\sigma))] = \sigma^\ell c.$$

The expected trajectory length depends exponentially on the depth!

Some Final Thoughts...

Expressivity:

- ▶ The goal is to bound the *approximation error*.
- ▶ Deep neural networks have a *universality property*.
- ▶ Function classes, which are optimal representable by *wavelets, shearlets, etc.*, are also optimally approximated by memory-efficient neural networks.
- ▶ *Deep* ReLU neural networks are *exponentially more efficient* in generating affine linear pieces.
- ▶ There are *alternative approaches* such as considering the length of the trajectory.

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*.

Related Book:

- Grohs and K, eds.,
Mathematical Aspects of Deep Learning
Cambridge University Press, 2022.

