MATHEMATICS OF DEEP LEARNING

RAJA GIRYES

TEL AVIV UNIVERSITY

Mathematics of Deep Learning Tutorial

CVPR, Hawaii

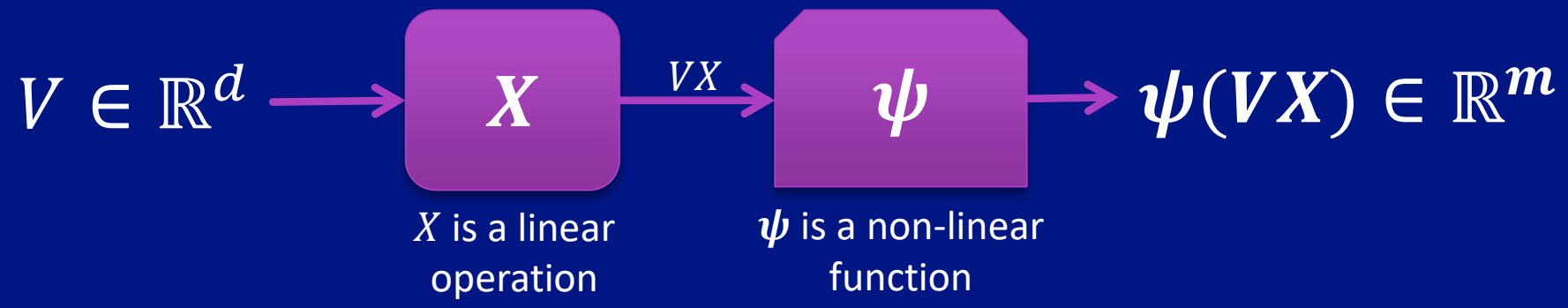
July 21, 2017

CUTTING EDGE PERFORMANCE IN MANY OTHER APPLICATIONS

- Disease diagnosis [Zhou, Greenspan & Shen, 2016].
- Language translation [Sutskever et al., 2014].
- Video classification [Karpathy et al., 2014].
- Handwriting recognition [Poznanski & Wolf, 2016].
- Sentiment classification [Socher et al., 2013].
- Image denoising [Remez et al., 2017].
- Depth Reconstruction [Haim et al., 2017].
- Super-resolution [Kim et al., 2016], [Bruna et al., 2016].
- many other applications…

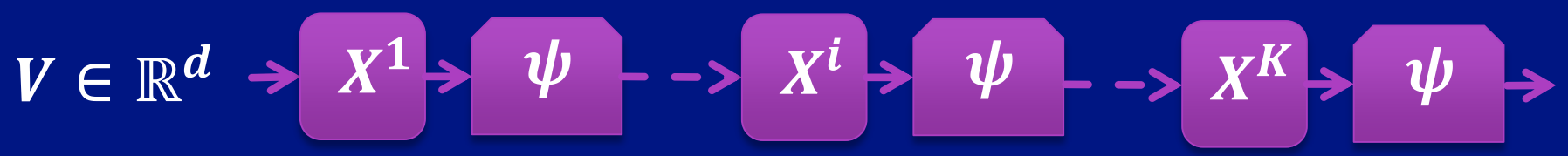
DEEP NEURAL NETWORKS (DNN)

- One layer of a neural net

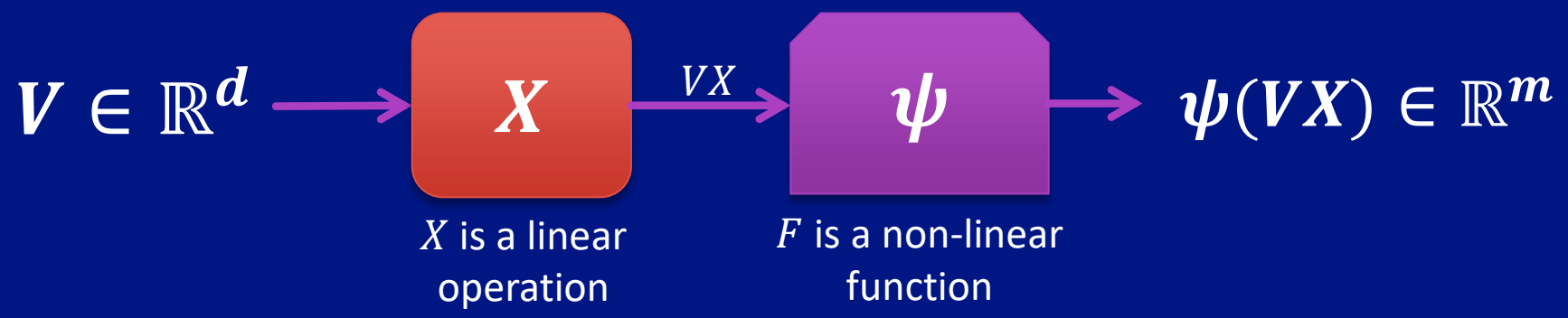


- Concatenation of the layers creates the whole net

$$\Phi(X^1, X^2, \dots, X^K) = \psi(\psi(\psi(VX^1)X^2) \dots X^K)$$



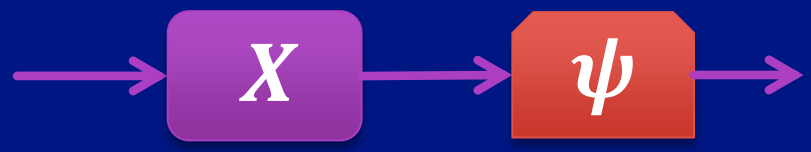
CONVOLUTIONAL NEURAL NETWORKS (CNN)



- In many cases, X is selected to be a convolution.
- This operator is shift invariant.
- CNN are commonly used with images as they are typically shift invariant.

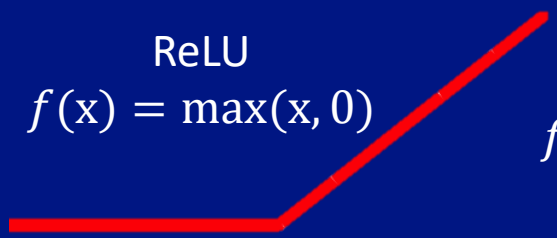
THE NON-LINEAR PART

- Usually $\psi = g \circ f$.

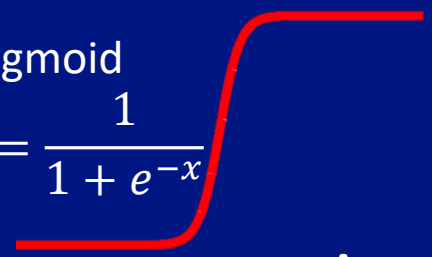


- f is the (point-wise) activation function

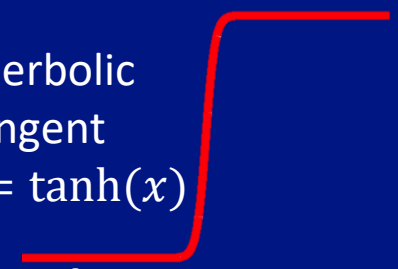
ReLU
$f(\mathrm{x}) = \max(\mathrm{x}, 0)$



Sigmoid
$f(x) = \dfrac{1}{1 + e^{-x}}$



Hyperbolic tangent
$f(x) = \tanh(x)$



- g is a pooling or an aggregation operator.

Max pooling

$\max_i V_i$

Mean pooling

$\frac{1}{n}\sum_{i=1}^{n} V_i$

l_p pooling

$\sqrt[p]{\sum_{i=1}^{n} V_i^p}$

WHY DNN WORK?

What is so special with the DNN structure?

What is the capability of DNN?

What is the role of the depth of DNN?

How many training samples do we need?

What is the role of the activation function?

What is the role of pooling?

What happens to the data throughout the layers?

OUTLINE

DNN keep the important information of the data.

Gaussian mean width is a good measure for the complexity of the data.

Important goal of training: Classify the boundary points between the different classes in the data.

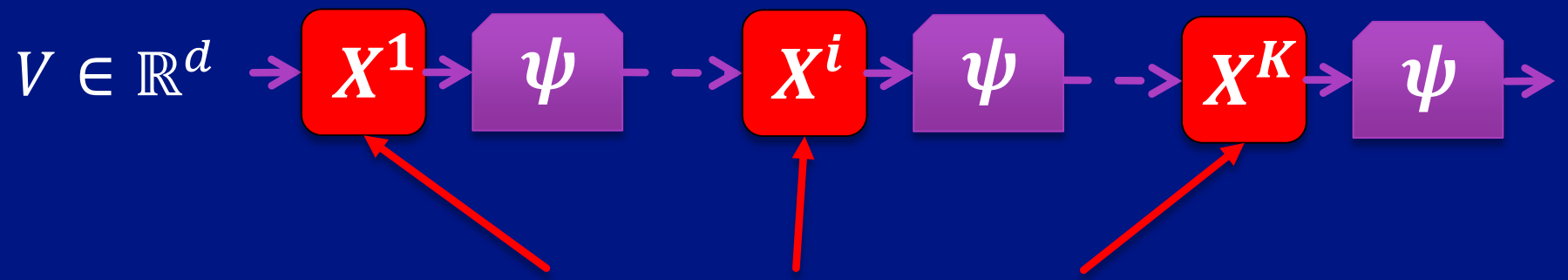
Random Gaussian weights are good for classifying the average points in the data.

DNN may solve optimization problems

Deep learning can be viewed as a metric learning.

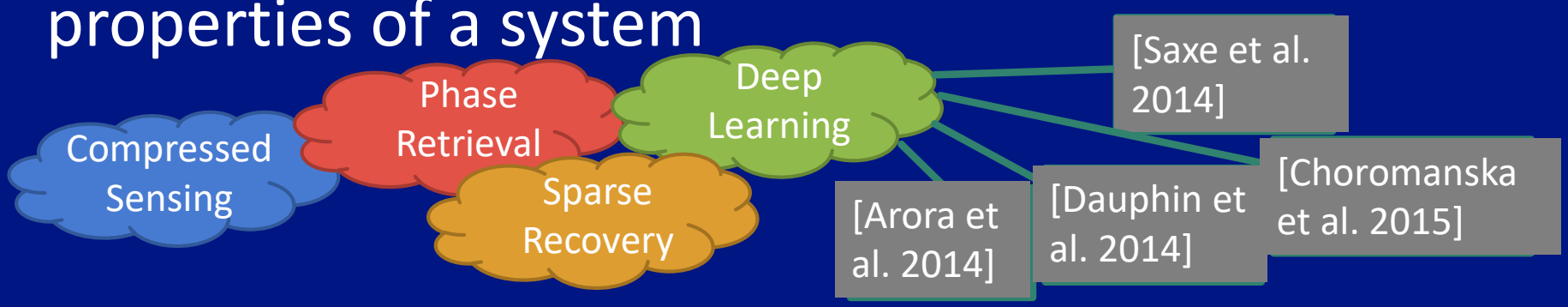
Generalization error depends on the DNN input margin

ASSUMPTIONS – GAUSSIAN WEIGHTS

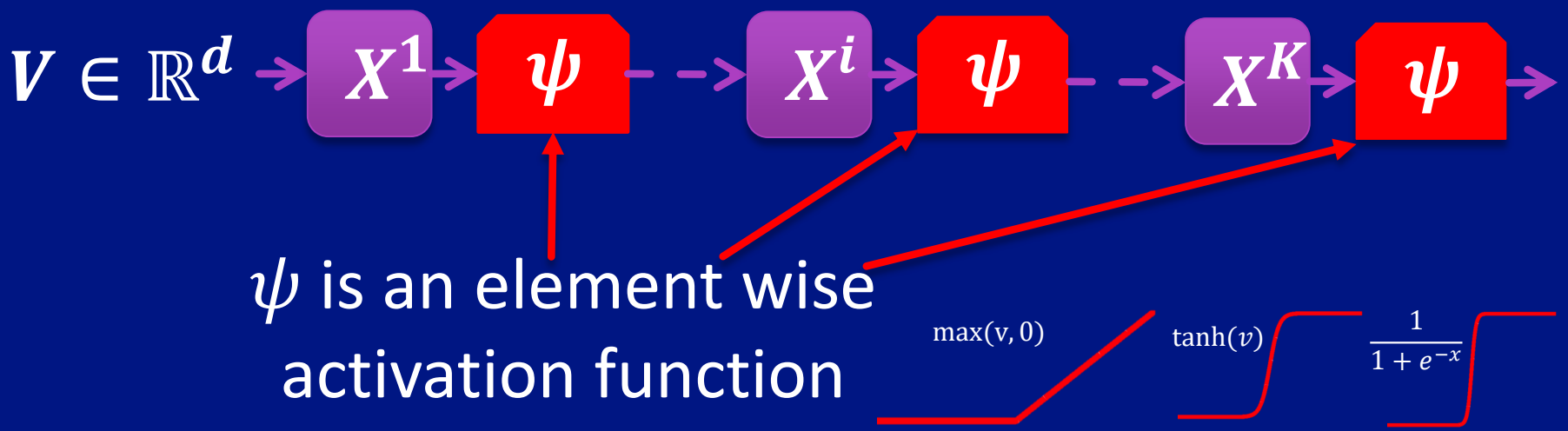


$X^i, \ldots, X^i, \ldots, X^K$ are random Gaussian matrices

- Infusion of random weights reveals internal properties of a system



ASSUMPTIONS – NO POOLING



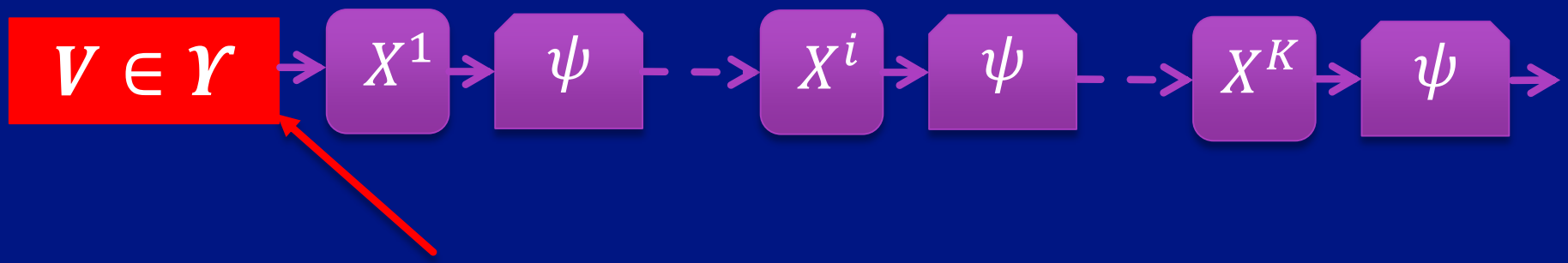
ψ is an element wise activation function

- Pooling provides invariance [Boureau et. al. 2010, Bruna et. al. 2013].

➢ We assume that all equivalent points in the data were merged together and omit this stage.

➢ Reveals the role of the other components in the DNN.

ASSUMPTIONS – LOW DIMENSIONAL DATA



Υ is a low dimensional set

Gaussian Mixture Models (GMM)

Low Rank Matrices

Signals with Sparse Representations

Low Dimensional Manifolds

Gaussian Mean Width

DNN keep the important information of the data.

Gaussian mean width is a good measure for the complexity of the data.

Important goal of training: Classify the boundary points between the different classes in the data.

Random Gaussian weights are good for classifying the average points in the data.

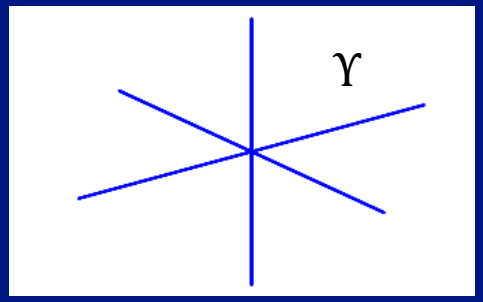
DNN may solve optimization problems

Deep learning can be viewed as a metric learning.

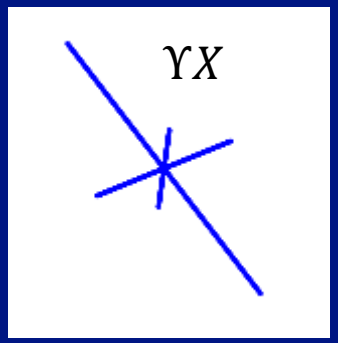
Generalization error depends on the DNN input margin

WHAT HAPPENS TO SPARSE DATA IN DNN?

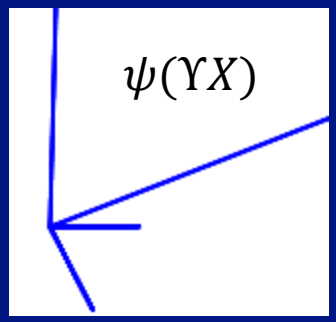
- Let Υ be sparsely represented data
  - Example: $\Upsilon = \{V \in \mathbb{R}^3 : \|V\|_0 \leq 1\}$



- $\Upsilon\mathrm{X}$ is still sparsely represented data
  - Example: $\Upsilon\mathrm{X} = \{V \in \mathbb{R}^3 : \exists W \in \mathbb{R}^3, V = WX, \|W\|_0 \leq 1\}$



- $\psi(\Upsilon\mathrm{X})$ not sparsely represented
- But is still low dimensional

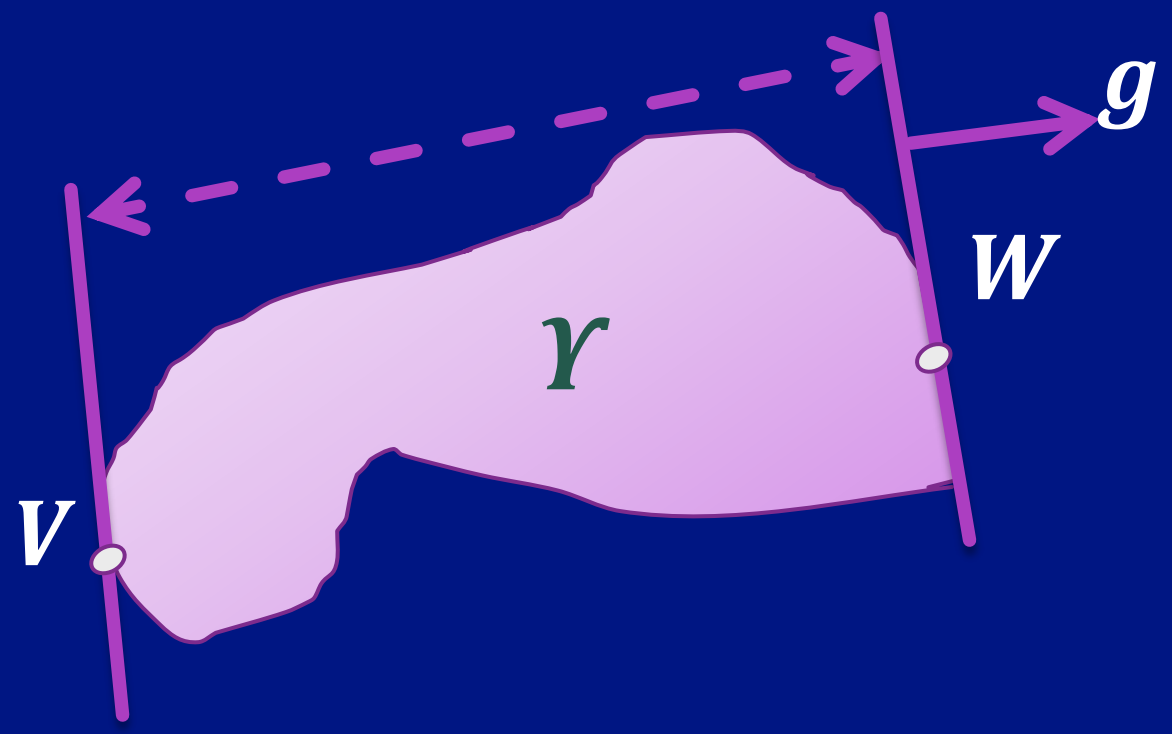


GAUSSIAN MEAN WIDTH

- Gaussian mean width:

$$\omega(\Upsilon) = E \sup_{V,W \in \Upsilon} \langle V - W, g \rangle, \qquad g \sim N(0, I).$$

The width of the set Υ in the direction of g:



MEASURE FOR LOW DIMENSIONALITY

- Gaussian mean width:

$$\boldsymbol{\omega(\Upsilon) = E \sup_{V,W\in\Upsilon} \langle V - W, g\rangle, \qquad g\sim N(0, I).}$$

- $\boldsymbol{\omega^2(\Upsilon)}$ is a measure for the dimensionality of the data.
- Examples:

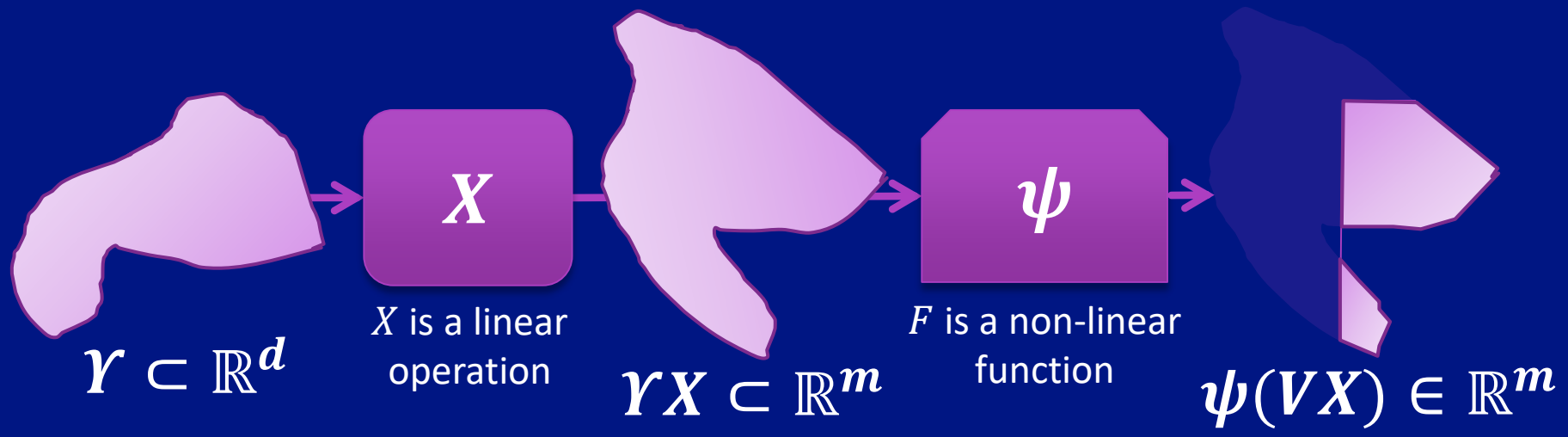
If $\Upsilon \subset \mathbb{B}^d$ is a Gaussian Mixture Model with k Gaussians then

$$\boldsymbol{\omega^2(\Upsilon) = O(k)}$$

If $\Upsilon \subset \mathbb{B}^d$ is a data with k-sparse representations then

$$\boldsymbol{\omega^2(\Upsilon) = O(k \log d)}$$

GAUSSIAN MEAN WIDTH IN DNN



Theorem 1: small $\frac{\omega^2(\Upsilon)}{m}$ imply $\omega^2(\Upsilon) \approx \omega^2(\psi(VX))$

Small $\omega^2(\Upsilon)$ → Small $\omega^2(\psi(VX))$

→ It is sufficient to provide proofs only for a single layer

Stability

DNN keep the important information of the data.

Gaussian mean width is a good measure for the complexity of the data.

Important goal of training: Classify the boundary points between the different classes in the data.

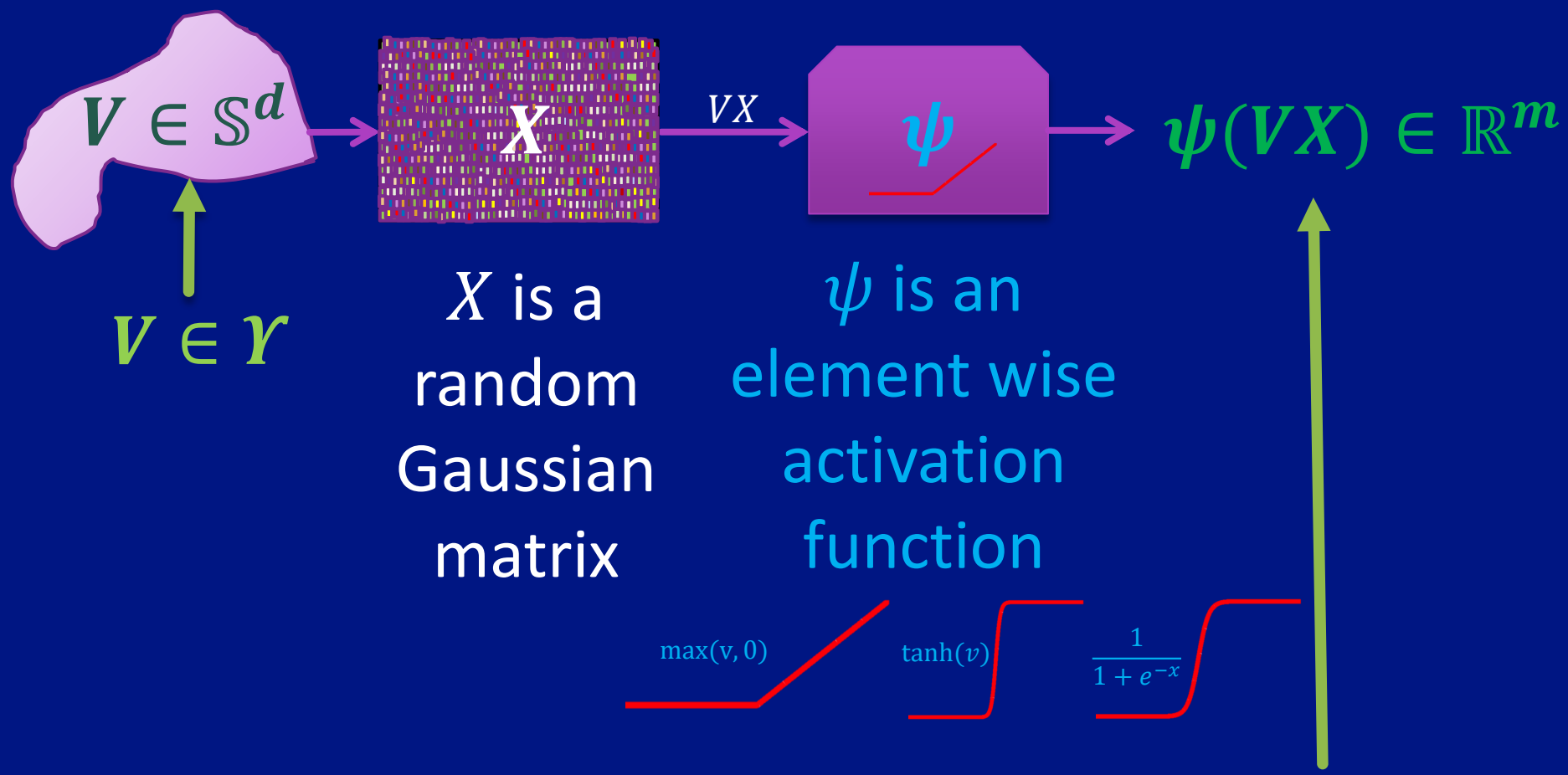
Random Gaussian weights are good for classifying the average points in the data.

DNN may solve optimization problems

Deep learning can be viewed as a metric learning.

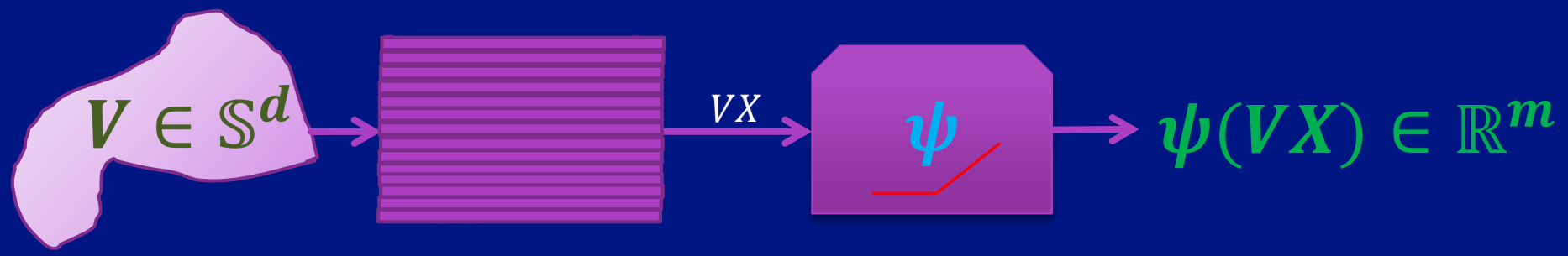
Generalization error depends on the DNN input margin

ASSUMPTIONS

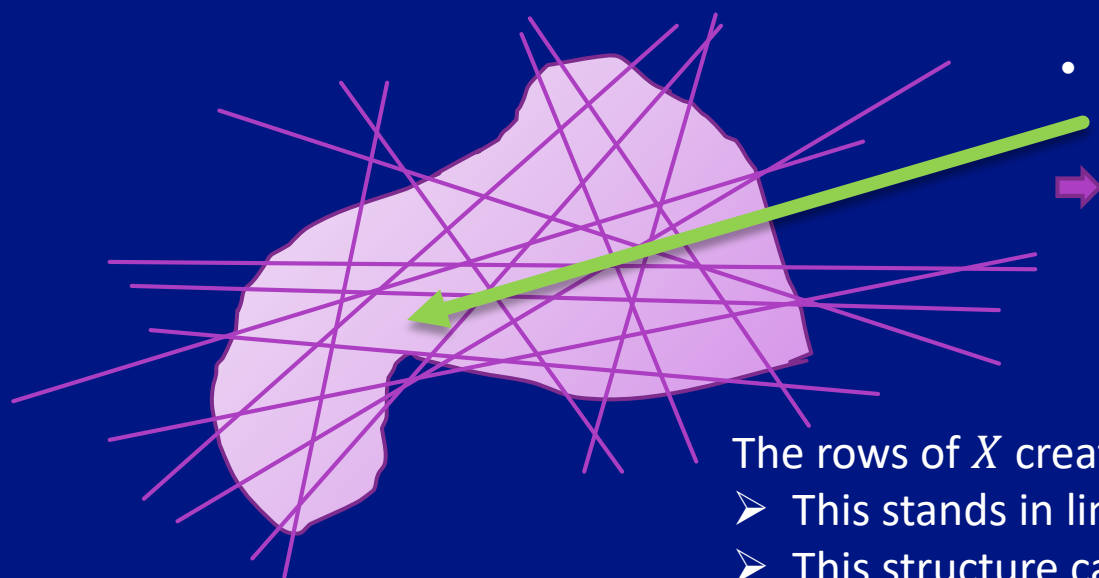


$$m = O\left(\delta^{-6}\omega^2(\Upsilon)\right)$$

ISOMETRY IN A SINGLE LAYER



Theorem 2: $\psi(\cdot\, X)$ is a δ-isometry in the Gromov-Hausdorff sense between the sphere $\mathbb{S}^{d-1}$ and the Hamming cube [Plan & Vershynin, 2014, Giryes, Sapiro & Bronstein 2016].



- If two points belong to the same tile then their distance $< \delta$

➡ Each layer of the network keeps the main information of the data

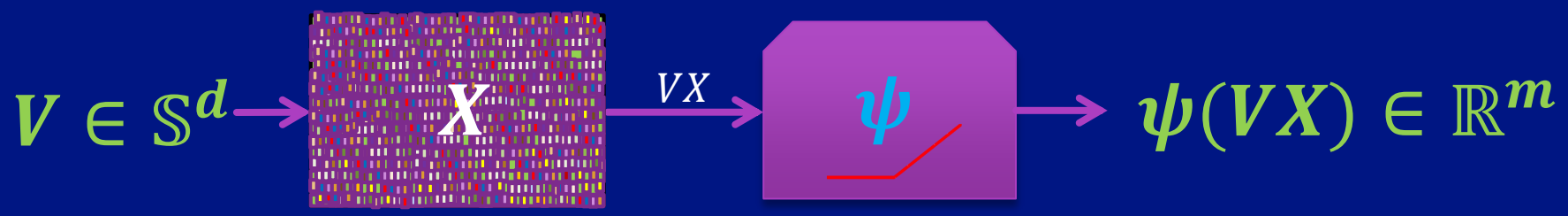
The rows of X create a tessellation of the space.

- ➢ This stands in line with [Montúfar et. al. 2014]
- ➢ This structure can be used for hashing

DNN AND HASHING

- A single layer performs a locally sensitive hashing.
- Deep network with random weights may be designed to do better [Choromanska et al., 2016].
- It is possible to train DNN for hashing, which provides cutting-edge results [Masci et al., 2012], [Lai et al., 2015].

DNN STABLE EMBEDDING



Theorem 3: There exists an algorithm $\mathcal{A}$ such that

$$\|V - \mathcal{A}(\psi(VX))\| < O\left(\frac{\omega(\Upsilon)}{\sqrt{m}}\right) = O(\delta^3)$$

[Plan & Vershynin, 2013, Giryes, Sapiro & Bronstein 2016].

- After K layers we have an error $O(K\delta^3)$
- Stands in line with [Mahendran and Vedaldi, 2015].
- DNN keep the important information of the data

RECOVERY FROM DNN OUTPUT

[Mahendran and Vedaldi, 2015].

DNN with Gaussian Weights

DNN keep the important information of the data.

Gaussian mean width is a good measure for the complexity of the data.

Important goal of training: Classify the boundary points between the different classes in the data.

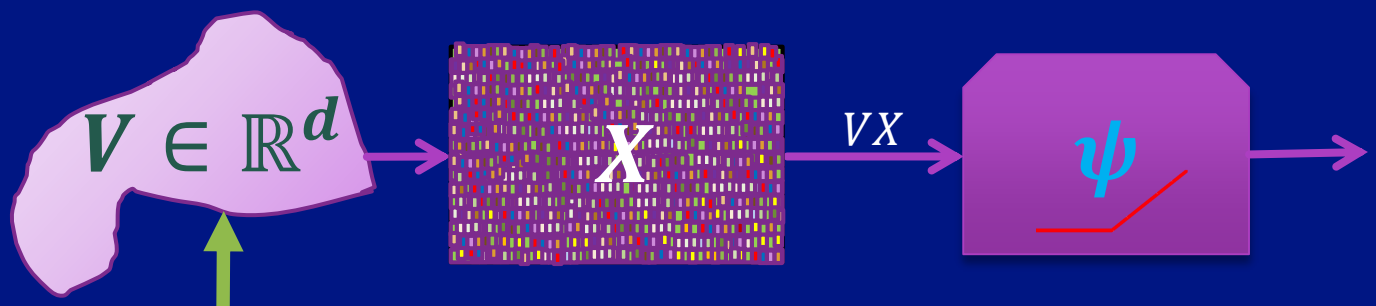
Random Gaussian weights are good for classifying the average points in the data.

DNN may solve optimization problems

Deep learning can be viewed as a metric learning.

Generalization error depends on the DNN input margin

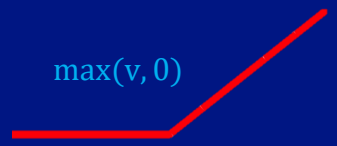
ASSUMPTIONS



$V \in \Upsilon$

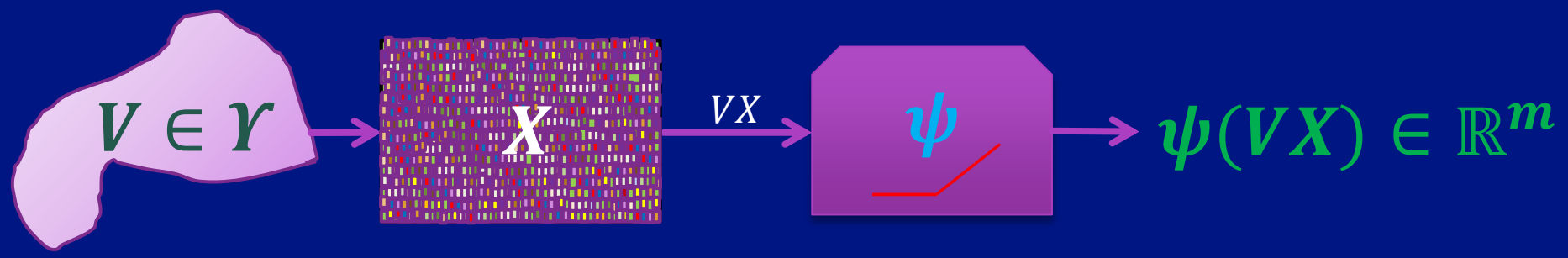
X is a random Gaussian matrix

ψ is the ReLU



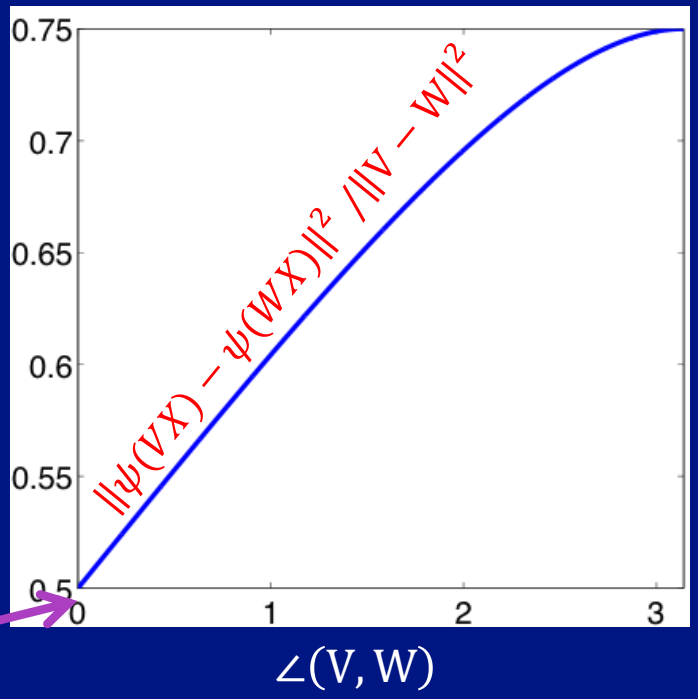
$$m = O\left(\delta^{-4}\omega^4(\Upsilon)\right)$$

DISTANCE DISTORTION



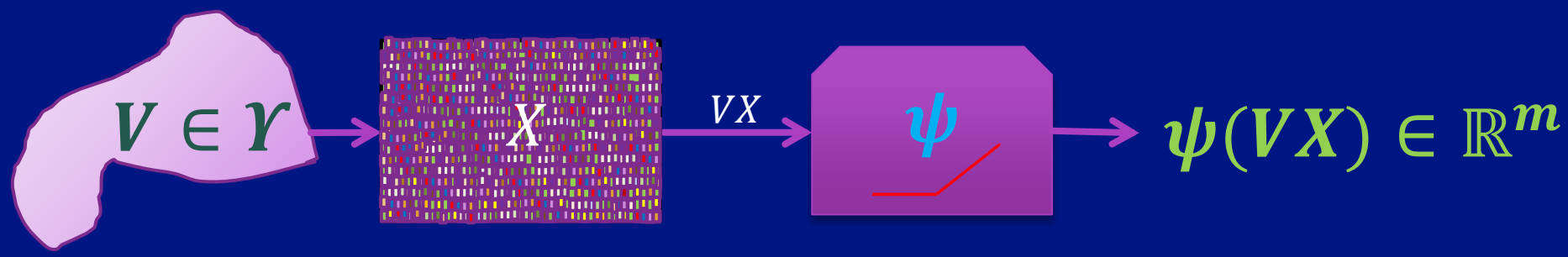
Theorem 4: for $V, W \in \Upsilon$

$$\left| \|\psi(VX) - \psi(WX)\|^2 - \frac{1}{2}\|V - W\|^2 - \frac{\|V\|\|W\|}{\pi}(\sin\angle(V,W)) \right|$$



The smaller $\angle(V,W)$ the smaller the distance we get between the points

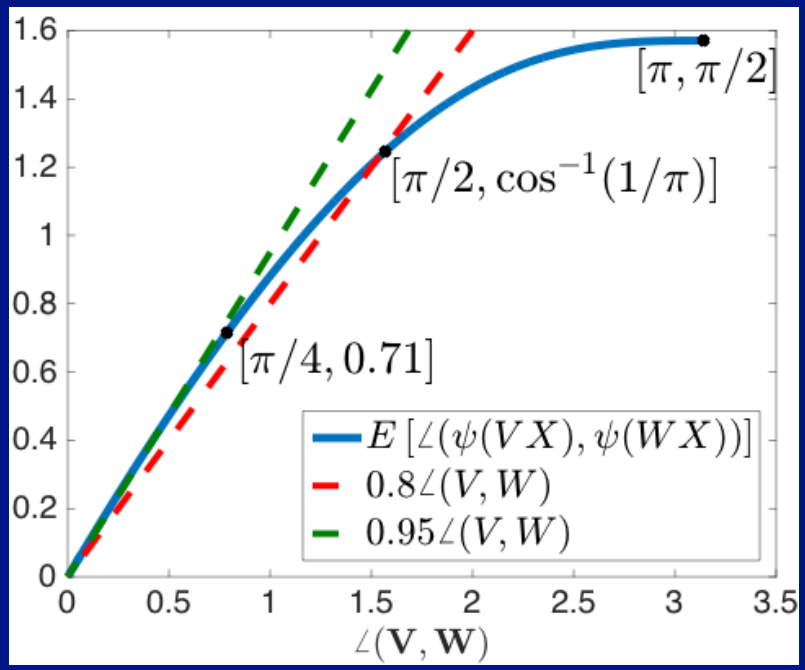
ANGLE DISTORTION



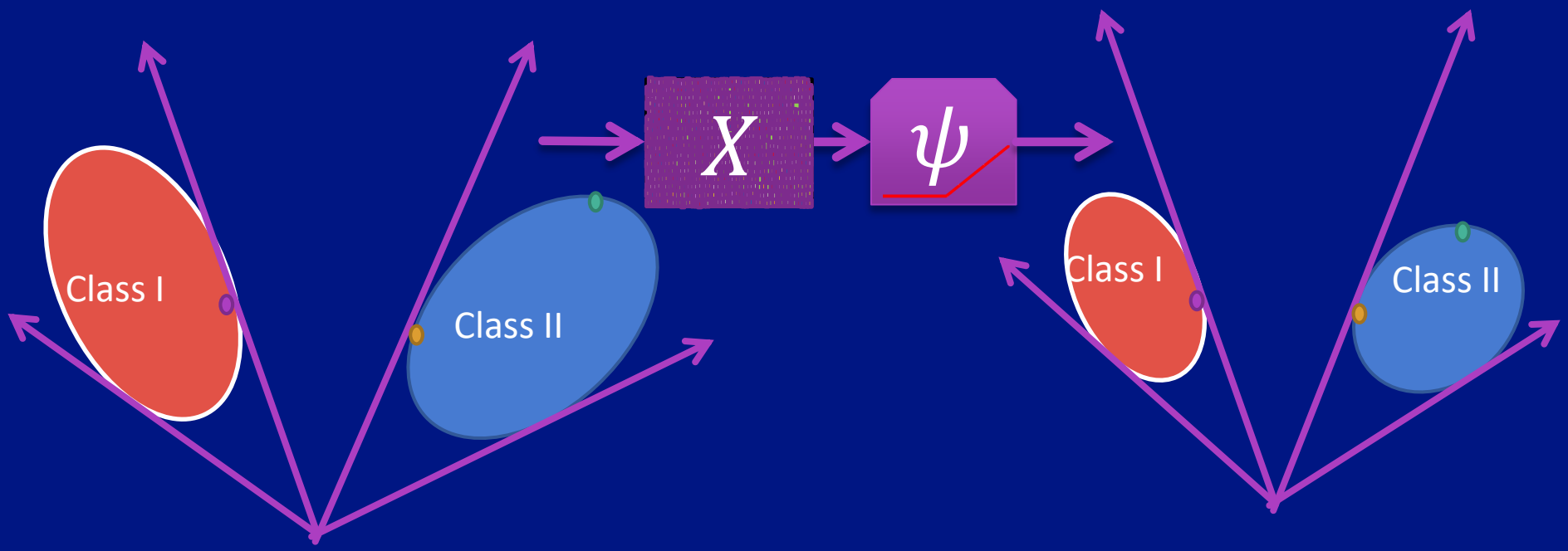
Theorem 5: for $V, W \in \Upsilon$

$$\left|\cos\angle(\psi(VX),\psi(WX)) - \cos\angle(V,W) - \frac{1}{\pi}(\sin\angle(V,W)\right|$$

Behavior of $\angle(\psi(VX),\psi(WX))$



DISTANCE AND ANGLES DISTORTION



Points with small angles between them become closer than points with larger angles between them

POOLING AND CONVOLUTIONS

- We test empirically this behavior on convolutional neural networks (CNN) with random weights and the MNIST, CIFAR-10 and ImageNet datasets.
- The behavior predicted in the theorems remains also in the presence of pooling and convolutions.

TRAINING DATA SIZE

- Stability in the network implies that close points in the input are close also at the output

➡ Having a good network for an ε-net of the input set Υ guarantees a good network for all the points in Υ.

➡ Using Sudakov minoration the number of data points is

$$\exp(\omega^2(\Upsilon)/\varepsilon^2)\,.$$

➡ Though this is not a tight bound, it introduces the Gaussian mean width $\omega\ (\Upsilon)$ as a measure for the complexity of the input data and the required number of training samples.

Role of Training

DNN keep the important information of the data.

Gaussian mean width is a good measure for the complexity of the data.

Important goal of training: Classify the boundary points between the different classes in the data.

Random Gaussian weights are good for classifying the average points in the data.

DNN may solve optimization problems

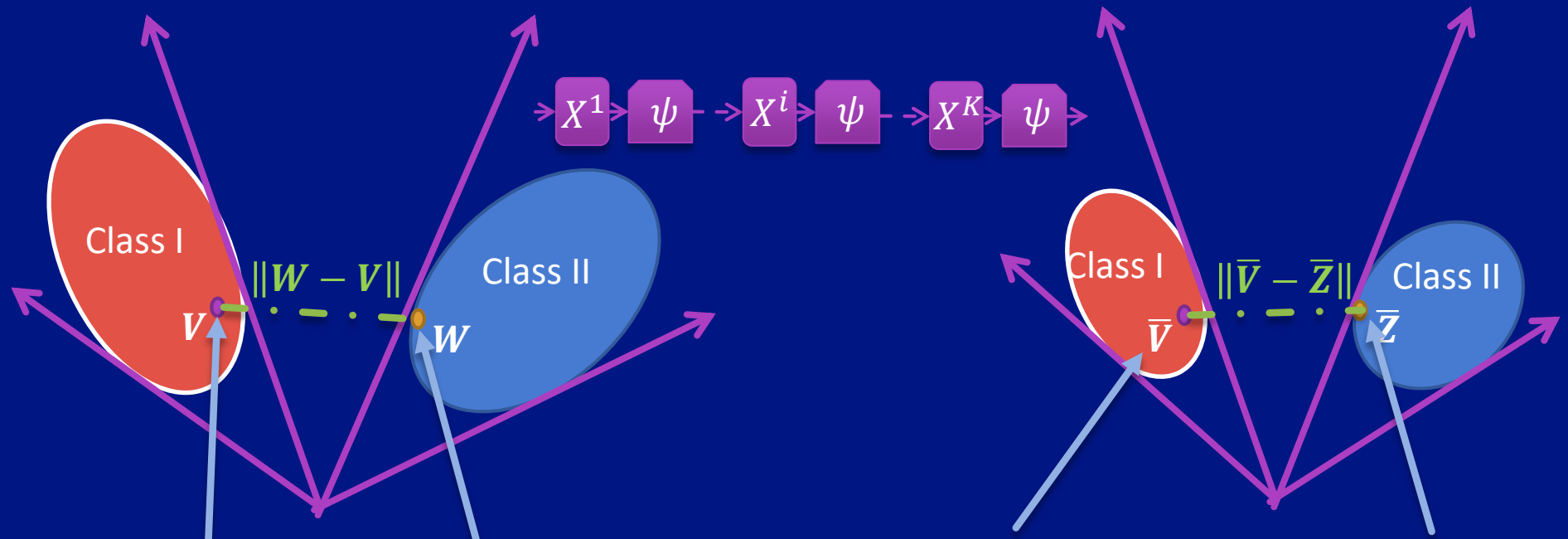
Deep learning can be viewed as a metric learning.

Generalization error depends on the DNN input margin

ROLE OF TRAINING

- Having a theory for Gaussian weights we test the behavior of DNN after training.
- We looked at the MNIST, CIFAR-10 and ImageNet datasets.
- We will present here only the ImageNet results.
- We use a state-of-the-art pre-trained network for ImageNet [Simonyan & Zisserman, 2014].
- We compute inter and intra class distances.

INTER BOUNDARY POINTS DISTANCE RATIO

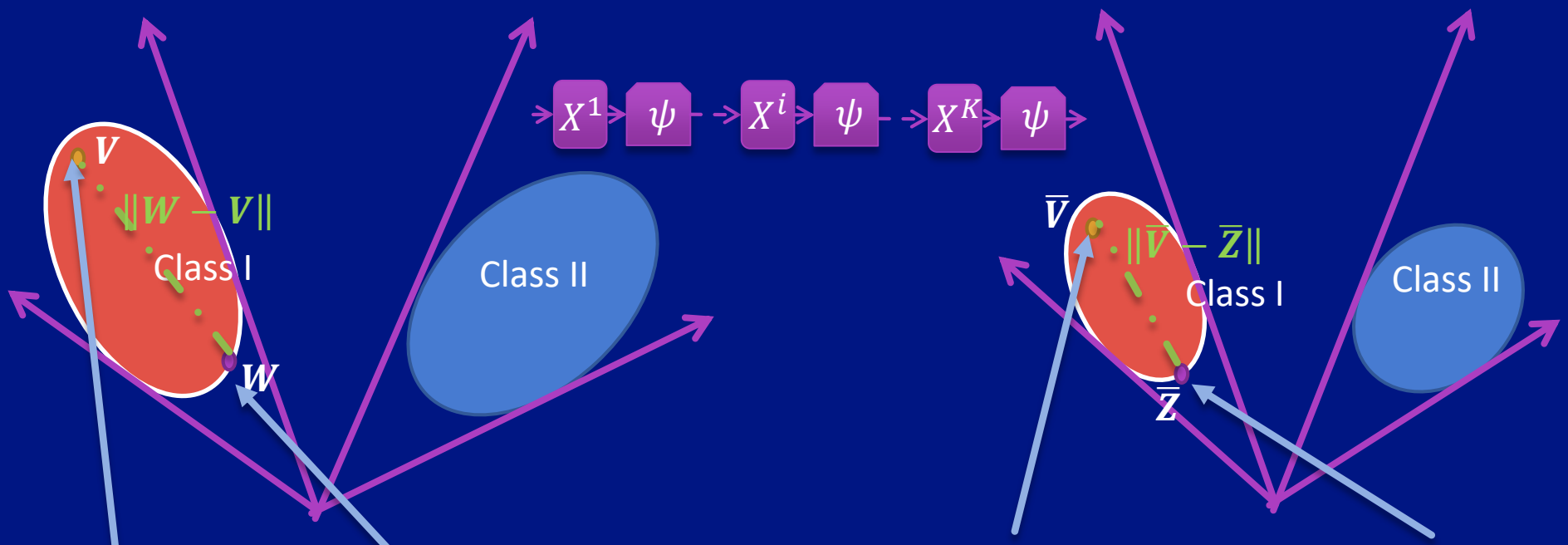


V is a random point and W its closest point from a different class.

$\bar{V}$ is the output of V and $\bar{Z}$ the closest point to $\bar{V}$ at the output from a different class.

Compute the distance ratio: $\frac{\|\bar{V}-\bar{Z}\|}{\|W-V\|}$

INTRA BOUNDARY POINTS DISTANCE RATIO



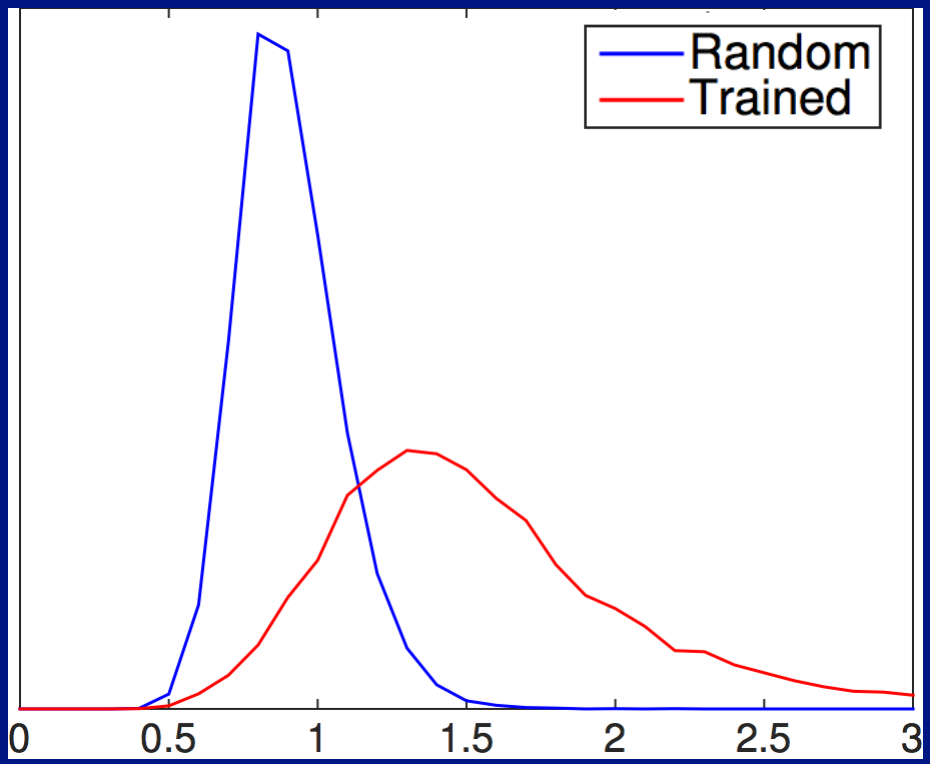
Let V be a point and W its farthest point from the same class.

Let $\bar{V}$ be the output of V and $\bar{Z}$ the farthest point from $\bar{V}$ at the output from the same class

Compute the distance ratio: $\frac{\|\bar{V}-\bar{Z}\|}{\|W-V\|}$

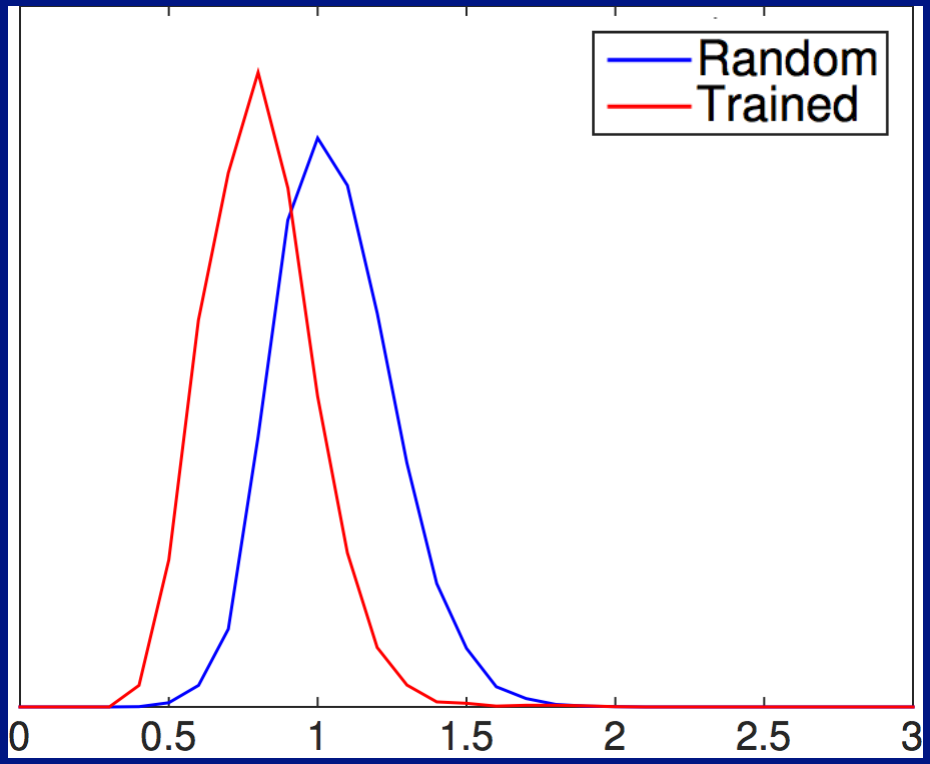
BOUNDARY DISTANCE RATIO

Inter-class



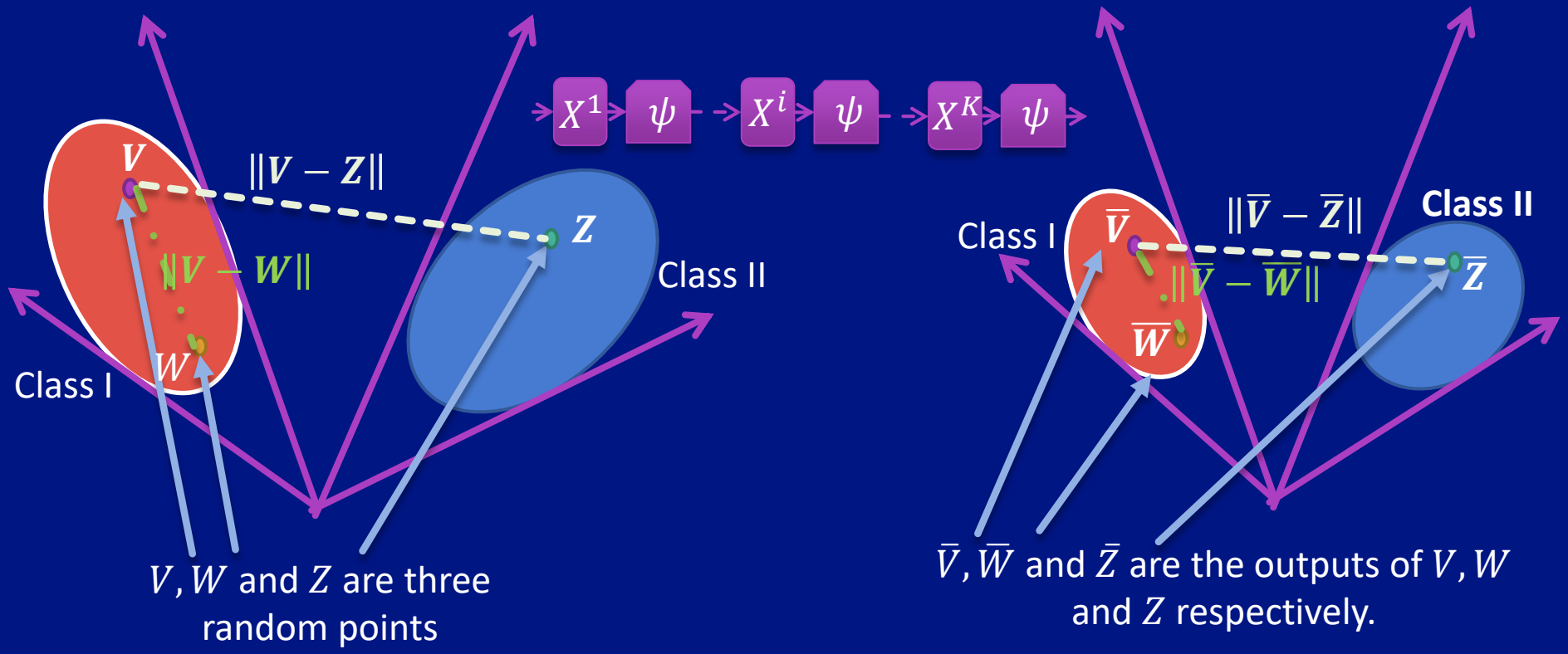
$$\frac{\|\bar{V} - \bar{Z}\|}{\|W - V\|}$$

Intra-class



$$\frac{\|\bar{V} - \bar{Z}\|}{\|W - V\|}$$

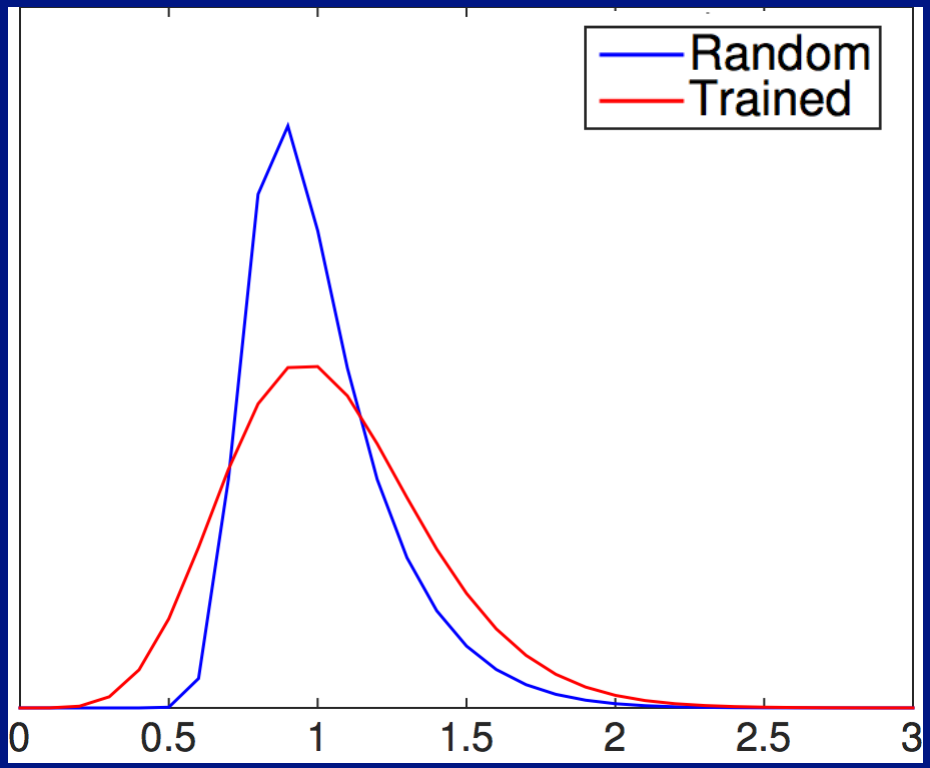
AVERAGE POINTS DISTANCE RATIO



Compute the distance ratios: $\frac{\|\bar{V}-\bar{W}\|}{\|V-W\|}, \frac{\|\bar{V}-\bar{Z}\|}{\|V-Z\|}$

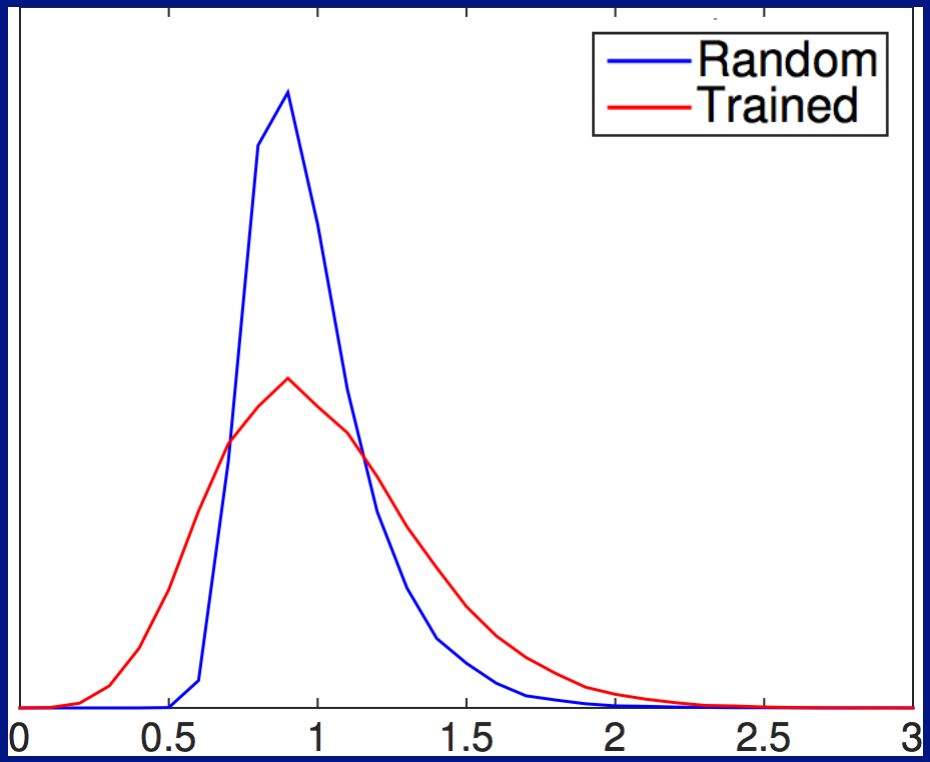
AVERAGE DISTANCE RATIO

Inter-class



$$\frac{\|\bar{V} - \bar{Z}\|}{\|V - Z\|}$$

Intra-class



$$\frac{\|\bar{V} - \bar{W}\|}{\|V - W\|}$$

ROLE OF TRAINING

- On average distances are preserved in the trained and random networks.
- The difference is with respect to the boundary points.
- The inter distances become larger.
- The intra distances shrink.

DNN as Metric Learning

DNN keep the important information of the data.

Gaussian mean width is a good measure for the complexity of the data.

Important goal of training: Classify the boundary points between the different classes in the data.

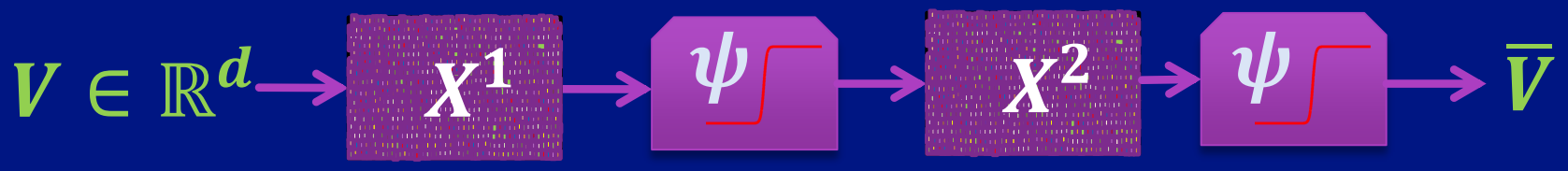
Random Gaussian weights are good for classifying the average points in the data.

DNN may solve optimization problems

Deep learning can be viewed as a metric learning.

Generalization error depends on the DNN input margin

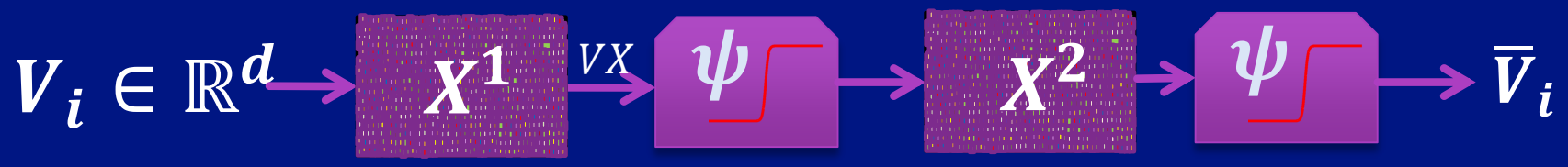
ASSUMPTIONS



X is fully connected and trained

ψ is the hyperbolic tan

METRIC LEARNING BASED TRAINING



- Cosine Objective:

$$\min_{X^1, X^2} \sum_{i,j \in Training\ Set} \left(\frac{\bar{V}_i^T \bar{V}_j}{\|\bar{V}_i\| \|\bar{V}_j\|} - \vartheta_{i,j} \right)^2$$

Classification term

Metric preservation term

$$\vartheta_{i,j} = \begin{cases} \lambda + (1-\lambda) \dfrac{V_i^T V_j}{\|V_i\| \|V_j\|} & i,j \in same\ class \\ -1 & i,j \in different\ clas \end{cases}$$

METRIC LEARNING BASED TRAINING

$$l_{ij} = \begin{cases} 1 & i,j \in \text{same class} \\ -1 & i,j \in \text{different class} \end{cases} \qquad l_{ij} = \begin{cases} \text{average intra class distance} & i,j \in \text{same class} \\ \text{average inter class distance} & i,j \in \text{different class} \end{cases}$$

- Euclidean Objective:

$$\min_{X^1,X^2} \frac{\lambda}{|\text{Training Set}|} \sum_{i,j \in \text{Training Set}} \left[l_{ij} \|\bar{V}_i - \bar{V}_j\| - t_{ij} \right]_+ + \frac{1-\lambda}{|\text{Neighbours}|} \sum_{V_i, V_j \text{ are neighbours}} \left| \|\bar{V}_i - \bar{V}_j\| - \|V_i - V_j\| \right|$$

Classification term

Metric learning term

ROBUSTNESS OF THIS NETWORK

- Metric learning objectives impose stability
- Similar to what we have in the random case
- Close points at the input are close at the output
- Using the theory of (T, ϵ)-robustness [Xu & Mannor, 2012], the generalization error scales as

$$\sqrt{\frac{T}{L}}$$

- T is the covering number and $L = |\text{Training set}|$.
- Also here, the number of training samples scales as

$$\exp(\omega^2(\Upsilon)/\varepsilon^2)\,.$$

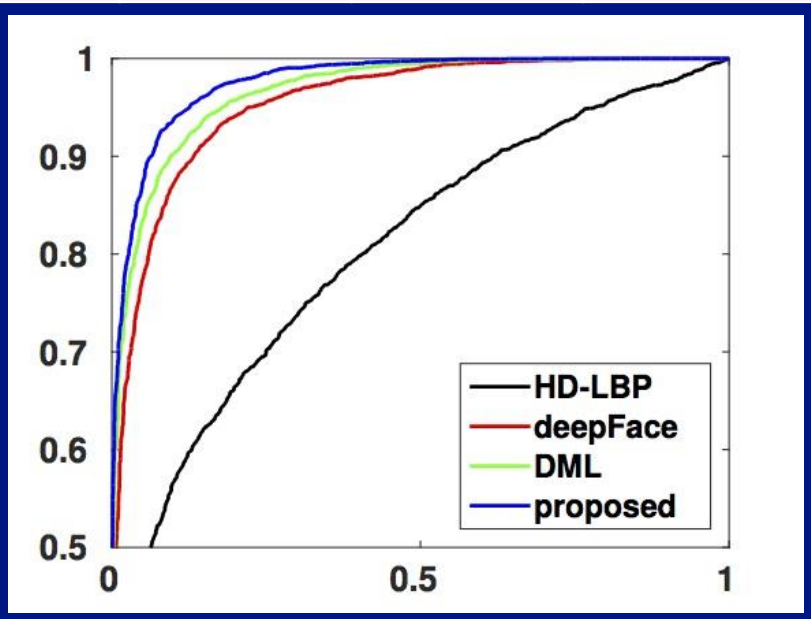
RESULTS

- Better performance with less training samples

MNIST Dataset

#Training/class	30	50	70	100
original pixels	81.91%	86.18%	86.86%	88.49%
LeNet	87.51%	89.89%	91.24%	92.75%
Proposed 1	92.32%	94.45%	95.67%	96.19%
Proposed 2	**94.14%**	**95.20%**	**96.05%**	**96.21%**

Faces in the wild ROC curve



[Huang, Qiu, Sapiro, Calderbank, 2015]

Generali-zation Error

DNN keep the important information of the data.

Gaussian mean width is a good measure for the complexity of the data.

Important goal of training: Classify the boundary points between the different classes in the data.

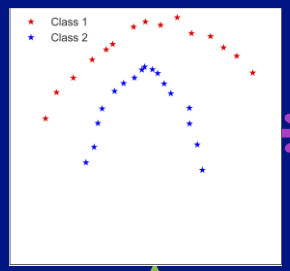
Random Gaussian weights are good for classifying the average points in the data.

DNN may solve optimization problems

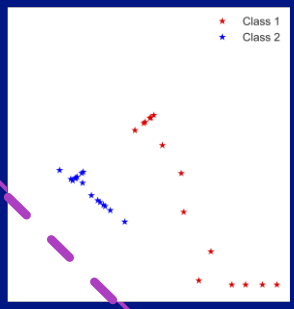
Deep learning can be viewed as a metric learning.

Generalization error depends on the DNN input margin

ASSUMPTIONS



X^1 → ψ → X^i → ψ → X^K → ψ

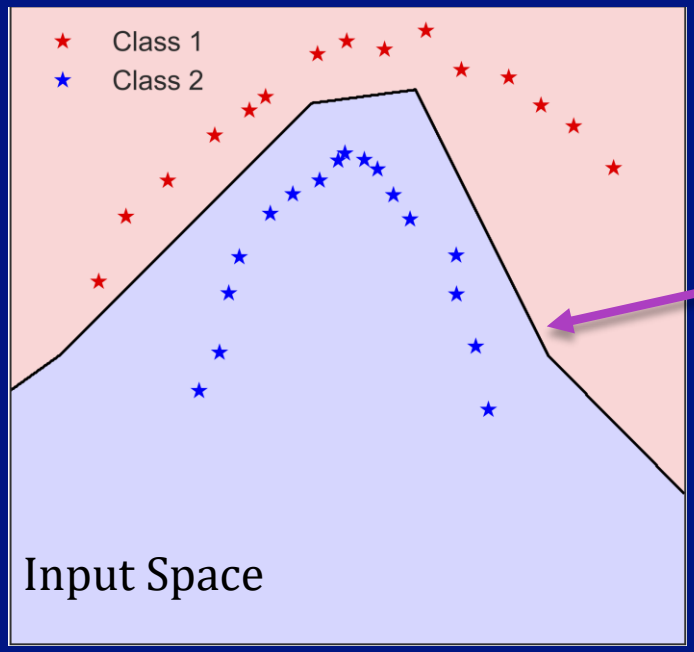


Two Classes $\in \mathcal{Y}$

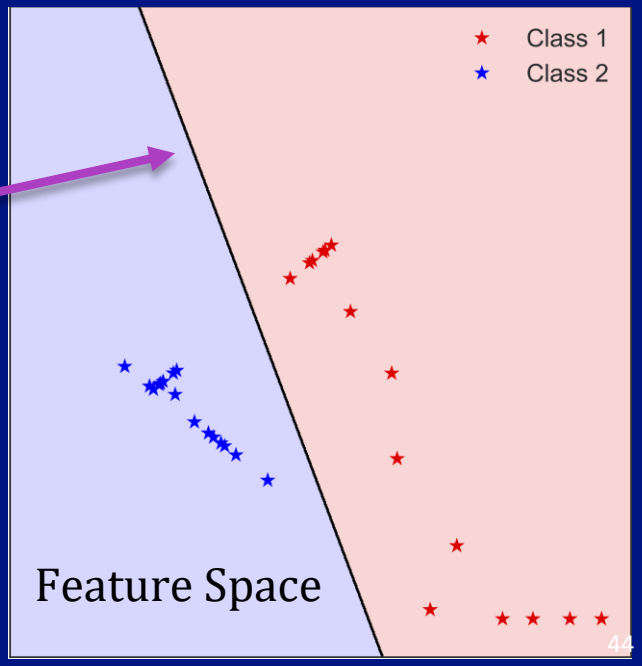
general non-linearity (ReLU, pooling,…)

softmax/ linear classifier

W



$$w^T \Phi(X^1, X^2, \ldots, X^K) = 0$$



CLASSIFIER TYPES

- Denote the output of the DNN by Z.
- Linear classifier W^T is of the form

$$ZW^T \underset{\text{Class 2}}{\overset{\text{Class 1}}{\gtrless}} b,$$

  where b is a certain threshold.
- Softmax classifier predicts the probability of class i:

$$\sigma(Z)_i = \frac{e^{Z_i}}{e^{Z_1} + e^{Z_2}}$$

CLASSIFICATION OBJETIVES

- Denote the output of the DNN by Z.
- Denote by t_i the expected output of Z_i
- Categorical cross entropy:

$$\sum \log(Z_i) t_i \,.$$

- Hinge loss:

$$\max(0,1 - Z_i t_i)$$

- Weight decay: penalty on the weight matrices,

$$\sum \|X^i\|$$

GENERALIZATION ERROR (GE)

- In training, we reduce the classification error ℓ_{training} of the training data as the number of training examples L increases.
- However, we are interested to reduce the error ℓ_{test} of the (unknown) testing data as L increases.
- The difference between the two is the generalization error

$$\text{GE} = \ell_{\text{training}} - \ell_{\text{test}}$$

⇨ It is important to understand the GE of DNN

ESTIMATION ERROR

- The estimation error of a function f by a neural networks scales as [Barron 1994].

$$O\left(\frac{C_f}{N}\right) + O\left(\frac{Nd}{L}\log(L)\right)$$

Smoothness of approximated function

Input dimension

Number of neurons in the DNN

Number of training examples

REGULARIZATION TECHNIQUES

- Weight decay – penalizing DNN weights [Krogh & Hertz, 1992].
- Dropout - randomly drop units (along with their connections) from the neural network during training [Hinton et al., 2012], [Baldi & Sadowski, 2013], Srivastava et al., 2014].
- DropConnect – dropout extension [Wan et al., 2013]
- Batch normalization [Ioffe & Szegedy, 2015].
- Stochastic gradient descent (SGD) [Hardt, Recht & Singer, 2016].
- Path-SGD [Neyshabur et al., 2015].
- And more [Rifai et al., 2011], [Salimans & Kingma, 2016], [Sun et al, 2016].

A SAMPLE OF GE BOUNDS – VC DIMENSION

- The VC dimension of a network with ReLUs is $O(K * \text{DNN params})$
- Thus,

$$\text{GE} \leq O\left(\sqrt{\text{DNN params} \cdot K \cdot \frac{\log(L)}{L}}\right)$$

[Bartlett et al. 1998, Shalev-Shwartz and Ben-David, 2014, Bartlett 2017, Harvey et al. 2017].

A SAMPLE OF GE BOUNDS – CAPACITY

- Bounding the GE using the capacity of the network:

$$\mathrm{GE} \leq \frac{1}{\sqrt{L}} 2^K \|w\|_2 \prod_i \|X^i\|_{2,2}$$

- Analysis relies on the Rademacher complexity of the network [Bartlett 1998, Neyshabur et al., 2015].

A SAMPLE OF GE BOUNDS

- Using the VC dimension it can be shown that

$$\mathrm{GE} \leq O\left(\sqrt{\text{DNN params} \cdot K \cdot \frac{\log(L)}{L}}\right)$$

[Bartlett et al. 1998, Shalev-Shwartz and Ben-David, 2014, Bartlett 2017, Harvey et al. 2017]

- The GE was bounded also by the DNN weights

$$\mathrm{GE} \leq \frac{1}{\sqrt{L}} 2^K \|w\|_2 \prod_i \|X^i\|_{2,2}$$

[Bartlett 1998, Neyshabur et al., 2015].

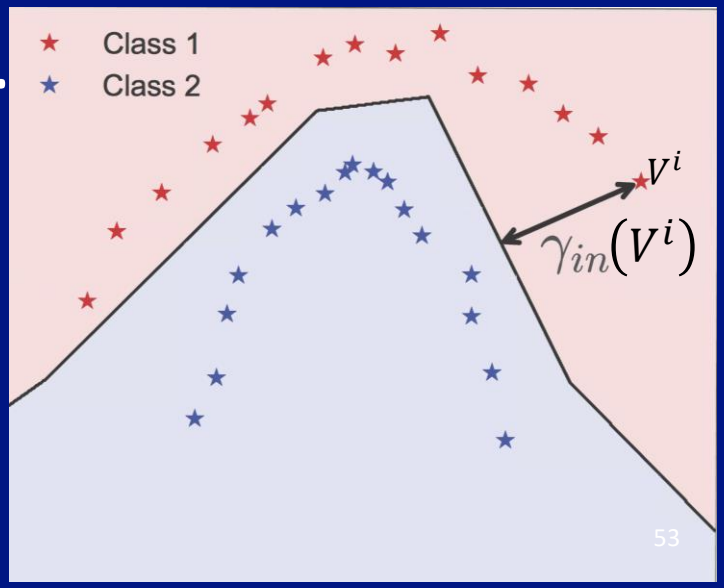
- Note that in both cases the GE grows with the depth

DNN INPUT MARGIN

- Theorem 6: If for every input margin $\gamma_{in}(V^i) > \gamma$

  then $$GE \leq \sqrt{N_{\gamma/2}(\Upsilon)}/\sqrt{L}$$

  [Sokolic, Giryes, Sapiro, Rodrigues, 2017]
- $N_{\gamma/2}(\Upsilon)$ is the covering number of the data Υ.
- $N_{\gamma/2}(\Upsilon)$ gets smaller as γ gets larger.
- Bound is independent of depth.
- Our theory relies on the robustness framework [Xu & Mannor, 2012].



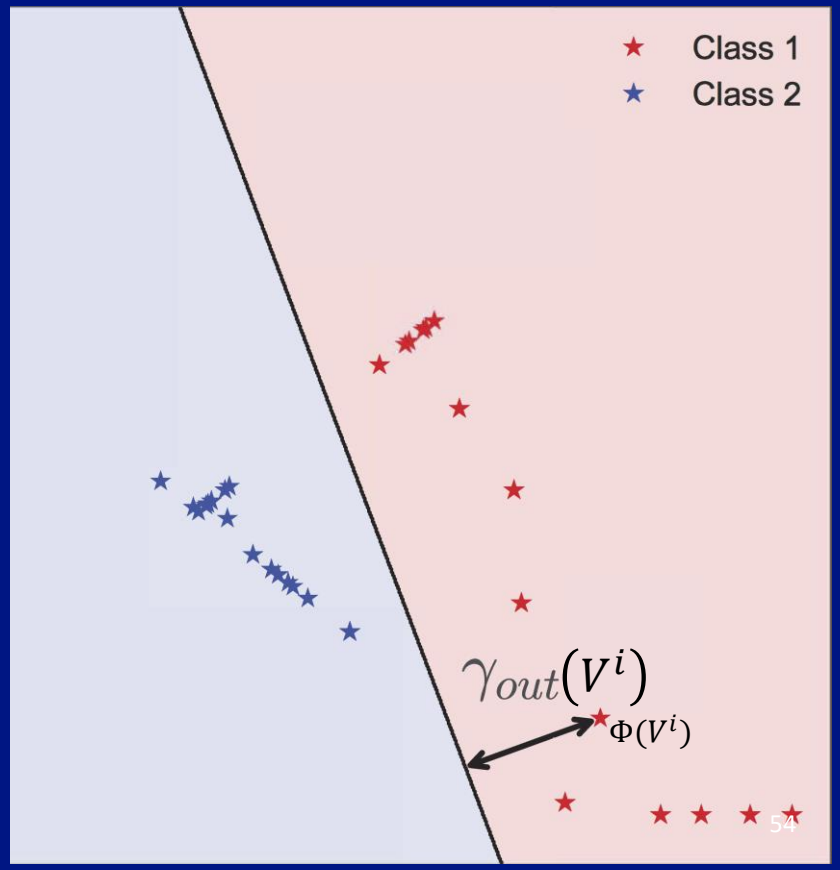
INPUT MARGIN BOUND

- Maximizing the input margin directly is hard
- Our strategy: relate the input margin to the output margin $\gamma_{out}(V^i)$ and other DNN properties
- Theorem 7:

$$\gamma_{in}(V^i) \geq \frac{\gamma_{out}(V^i)}{\sup_{V \in \Upsilon} \left\| \frac{V}{\|V\|_2} J(V) \right\|_2}$$

$$\geq \frac{\gamma_{out}(V^i)}{\prod_{1 \leq i \leq K} \|X^i\|_2}$$

$$\geq \frac{\gamma_{out}(V^i)}{\prod_{1 \leq i \leq K} \|X^i\|_F}$$

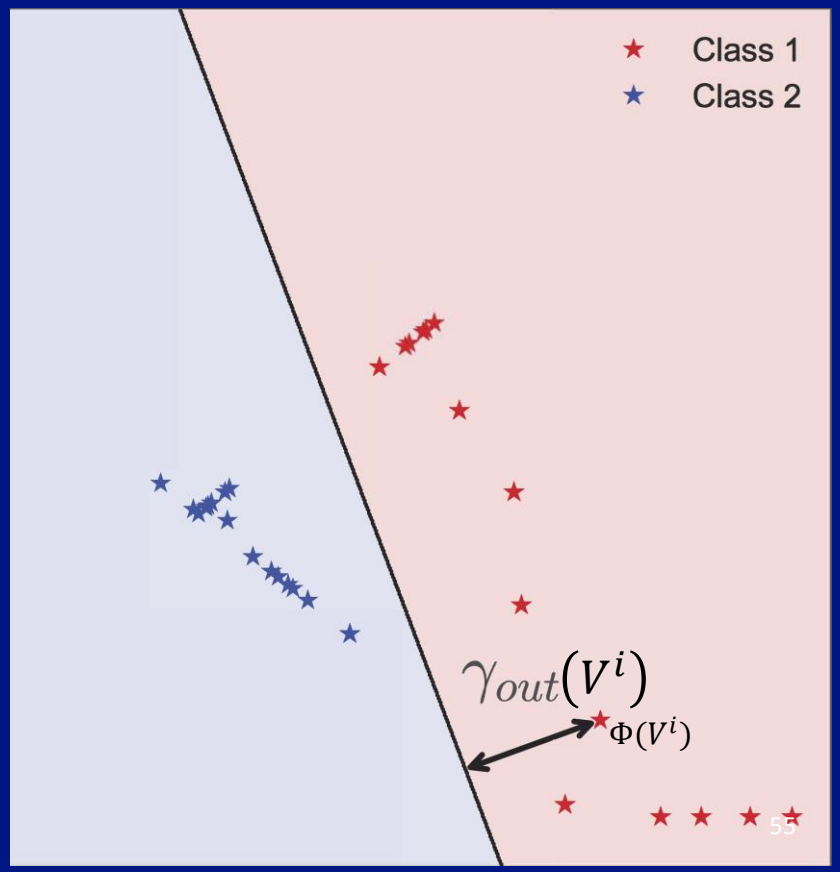


[Sokolic, Giryes, Sapiro, Rodrigues, 2017]

OUTPUT MARGIN

- Theorem 7: $\gamma_{in}(V^i) \geq \dfrac{\gamma_{out}(V^i)}{\sup\limits_{V \in \Upsilon} \left\| \frac{V}{\|V\|_2} J(V) \right\|_2}$

  $\geq \dfrac{\gamma_{out}(V^i)}{\prod_{1 \leq i \leq K} \|X^i\|_2} \geq \dfrac{\gamma_{out}(V^i)}{\prod_{1 \leq i \leq K} \|X^i\|_F}$
- Output margin is easier to maximize – SVM problem
- Maximized by many cost functions, e.g., hinge loss.

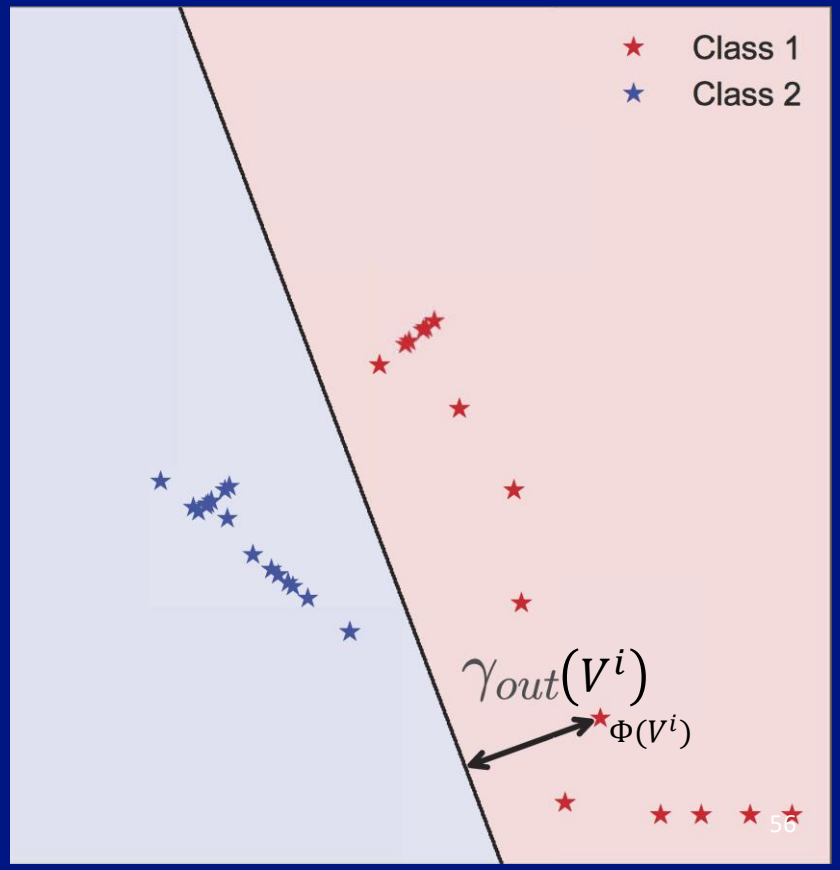


GE AND WEIGHT DECAY

- Theorem 7: $\gamma_{in}(V^i) \geq \frac{\gamma_{out}(V^i)}{\sup_{V \in \Upsilon} \left\| \frac{V}{\|V\|_2} J(V) \right\|_2} \geq \frac{\gamma_{out}(V^i)}{\prod_{1 \leq i \leq K} \|X^i\|_2}$

  $\geq \frac{\gamma_{out}(V^i)}{\prod_{1 \leq i \leq K} \|X^i\|_F}$

- Bounding the weights increases the input margin
- Weight decay regularization decreases the GE
- Related to regularization used by [Haeffele & Vidal, 2015]



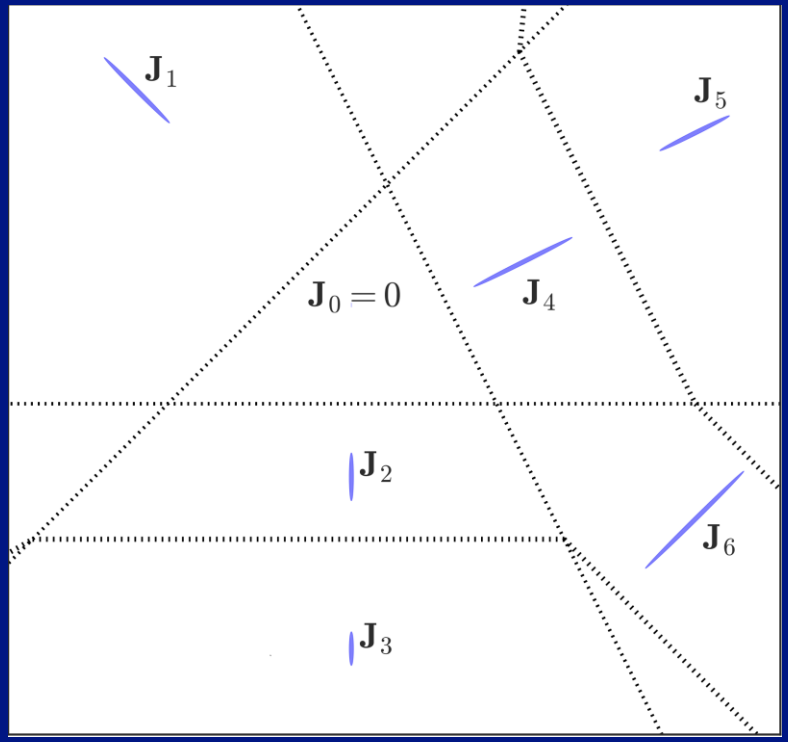
JACOBIAN BASED REGULARIZATION

- Theorem 7: $\gamma_{in}(V^i) \geq \frac{\gamma_{out}(V^i)}{\sup_{V \in \Upsilon} \left\| \frac{V}{\|V\|_2} J(V) \right\|_2} \geq \frac{\gamma_{out}(V^i)}{\prod_{1 \leq i \leq K} \|X^i\|_2}$

  $\geq \frac{\gamma_{out}(V^i)}{\prod_{1 \leq i \leq K} \|X^i\|_F}$

- $J(V)$ is the Jacobian of the DNN at point V.
- $J(\cdot)$ is piecewise constant.
- Using the Jacobian of the DNN leads to a better bound.

➡ New regularization technique.



RESULTS

- Better performance with less training samples

MNIST Dataset

		256 samples			512 samples			1024 samples		
loss	# layers	no reg.	WD	LM	no reg.	WD	LM	no reg.	WD	LM
hinge	2	88.37	89.88	**93.83**	93.99	94.62	95.49	95.79	96.57	97.45
hinge	3	87.22	89.31	93.22	93.41	93.97	**95.76**	95.46	96.45	**97.60**
CCE	2	88.45	88.45	92.77	92.29	93.14	95.25	95.38	95.79	96.89
CCE	3	89.05	89.05	93.10	91.81	93.02	95.32	95.11	95.86	97.14

- CCE: the categorical cross entropy.
- WD: weight decay regularization.
- LM: Jacobian based regularization for large margin.
- Note that hinge loss generalizes better than CCE and that LM is better than WD as predicted by our theory.

[Sokolic, Giryes, Sapiro, Rodrigues, 2017]

INVARIANCE

- Our theory extends also to study of the relation between invariance in the data and invariance in the network
- We have proposed also a new strategy to enforce invariance in the network [Sokolic, Giryes, Sapiro, Rodrigues, 2017]

INVARIANCE

- Designing invariant DNN reduce the GE

Table 1: Classification accuracy [%] on CIFAR-10.

	number of training samples				
	2500	5000	10000	20000	50000
No reg.	68.71	76.74	85.17	87.15	93.65
Inv. Reg.	69.32	79.08	86.69	88.14	94.50
No reg. + avg.	70.59	78.40	86.05	88.13	94.26
Inv. Reg. + avg.	70.71	79.65	86.96	88.98	94.78

[Sokolić, Giryes, Sapiro & Rodrigues, 2017]

Minimization by DNN

DNN keep the important information of the data.

Gaussian mean width is a good measure for the complexity of the data.

Important goal of training: Classify the boundary points between the different classes in the data.

Random Gaussian weights are good for classifying the average points in the data.

DNN may solve optimization problems

Deep learning can be viewed as a metric learning.

Generalization error depends on the DNN input margin

INVERSE PROBLEMS

- We are given $\boldsymbol{V} = \boldsymbol{Z}\boldsymbol{A} + \boldsymbol{E}$

Given set of measurements

Unknown signal

linear operator

noise

- Standard technique for recovery

$$\min_{\boldsymbol{Z}} \|\boldsymbol{V} - \boldsymbol{Z}\boldsymbol{A}\|_2 \qquad \mathbf{s.t.} \qquad \boldsymbol{Z} \in \boldsymbol{\Upsilon}$$

$\boldsymbol{Z}$ resides in a low dimensional set $\boldsymbol{\Upsilon}$

- Unconstrained form

$$\min_{\boldsymbol{Z}} \|\boldsymbol{V} - \boldsymbol{Z}\boldsymbol{A}\|_2^2 + \boldsymbol{\lambda} \boldsymbol{f}(\boldsymbol{Z})$$

Regularization parameter

$\boldsymbol{f}$ is a penalty function

ℓ_1 MINIMIZATION CASE

- Unconstrained form

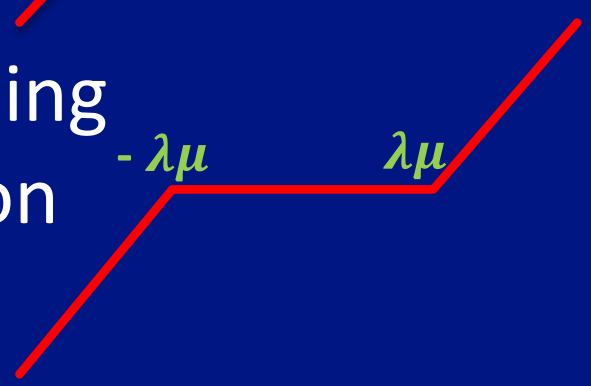
$$\min_{Z} \|V - ZA\|_2^2 + \lambda \|Z\|_1$$

- Can be solved by iterative shrinkage and thresholding technique (ISTA)

$$Z^{t+1} = \psi_{\lambda\mu}\left(Z^t + \mu(V - Z^t A)A^T\right)$$

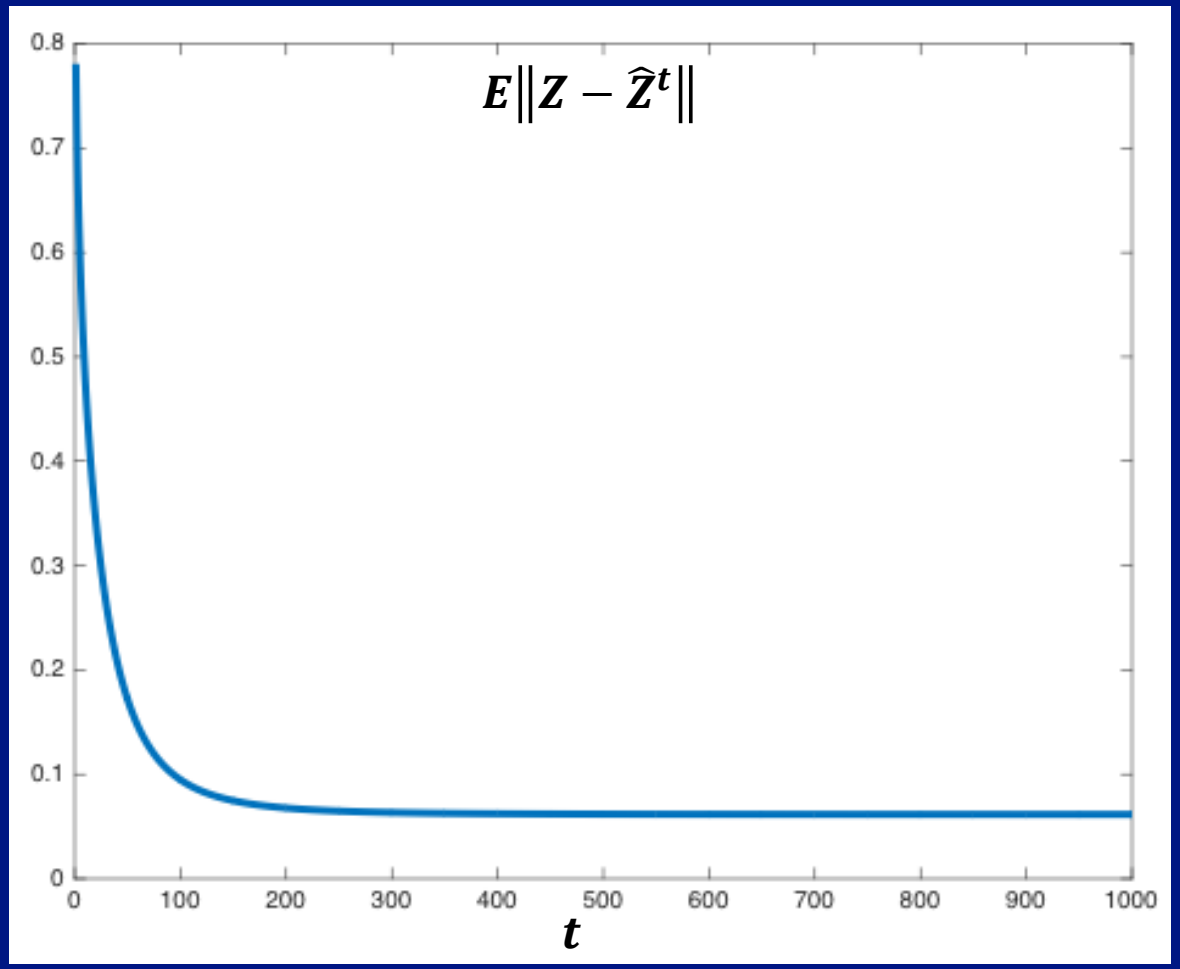
Soft thresholding operation

$\boldsymbol{\mu}$ is the step size



ISTA CONVERGENCE

- Reconstruction mean squared error (MSE) as a function of the number of iterations



$\boldsymbol{LISTA}$

- ISTA

$$\boldsymbol{Z^{t+1}} = \boldsymbol{\psi}_{\lambda\mu}\left(\boldsymbol{Z^t} + \mu(\boldsymbol{V} - \boldsymbol{Z^t A})\boldsymbol{A^T}\right)$$

- Rewriting ISTA:

$$\boldsymbol{Z^{t+1}} = \boldsymbol{\psi}_{\lambda\mu}\left(\boldsymbol{Z^t}\left(\boldsymbol{I} - \boldsymbol{\mu AA^T}\right) + \mu \boldsymbol{VA^T}\right)$$

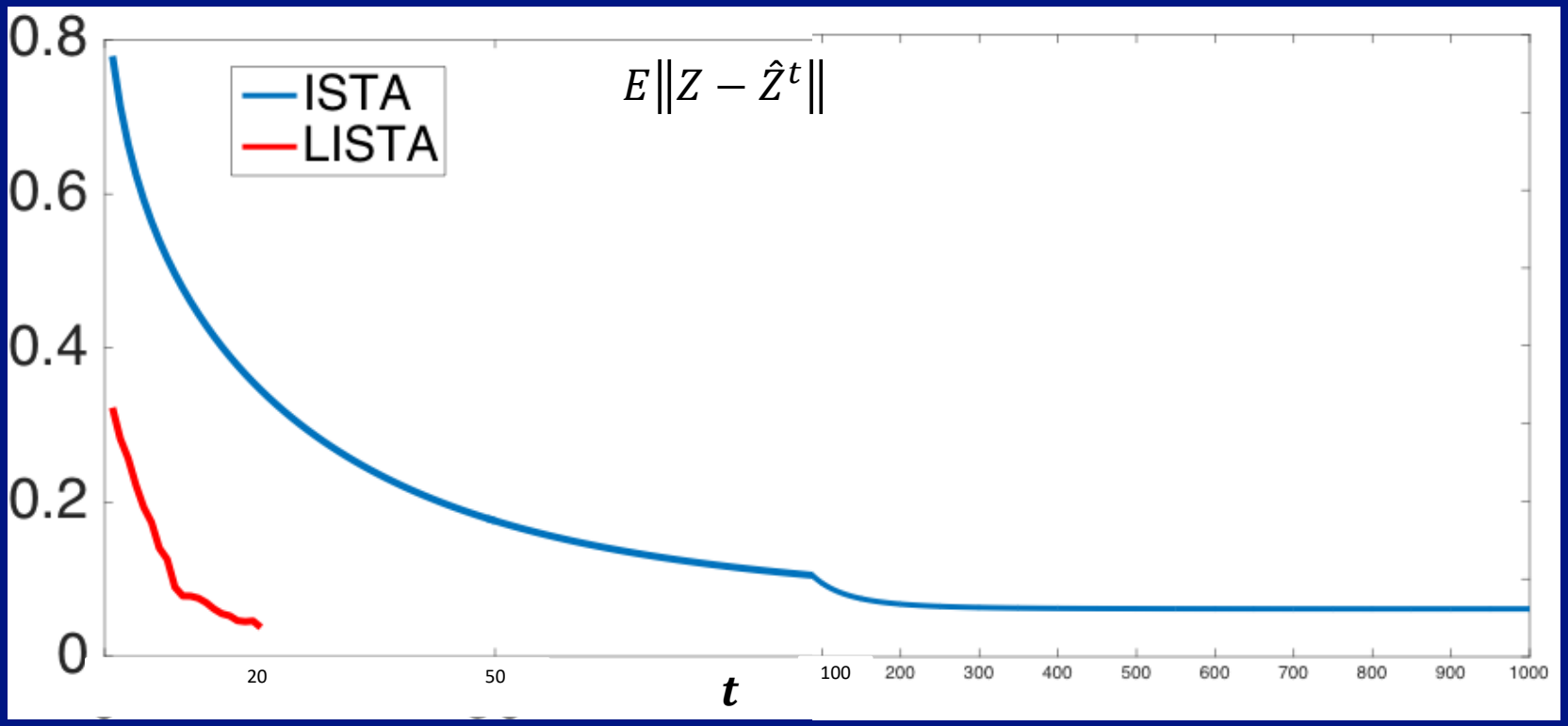
- Learned ISTA (LISTA):

$$\boldsymbol{Z^{t+1}} = \boldsymbol{\psi}_{\lambda}(\boldsymbol{Z^t X} + \boldsymbol{VS})$$

Learned operators

LISTA CONVERGENCE

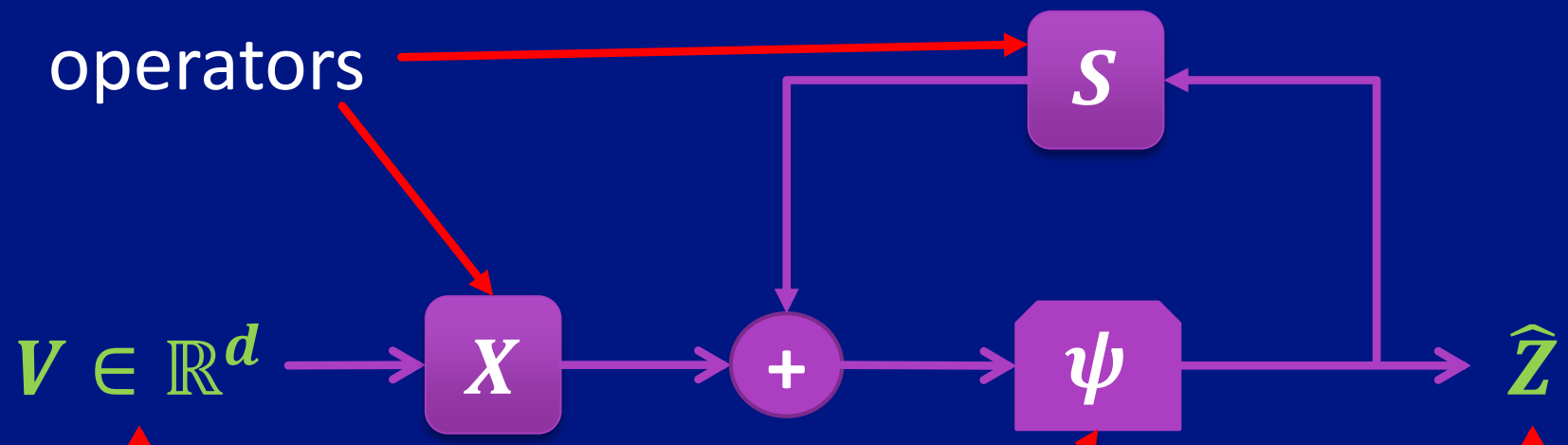
- Replacing $I - \mu AA^T$ and μA^T in ISTA with the learned X and S improves convergence [Gregor & LeCun, 2010]



- Extensions to other models [Sprechmann, Bronstein & Sapiro, 2015], [Remez, Litani & Bronstein, 2015], [Tompson, Schlachter, Sprechmann & Perlin, 2016].

LISTA AS A NEURAL NETWORK

Linear operators



$$V = ZA + E$$

linear operator

noise

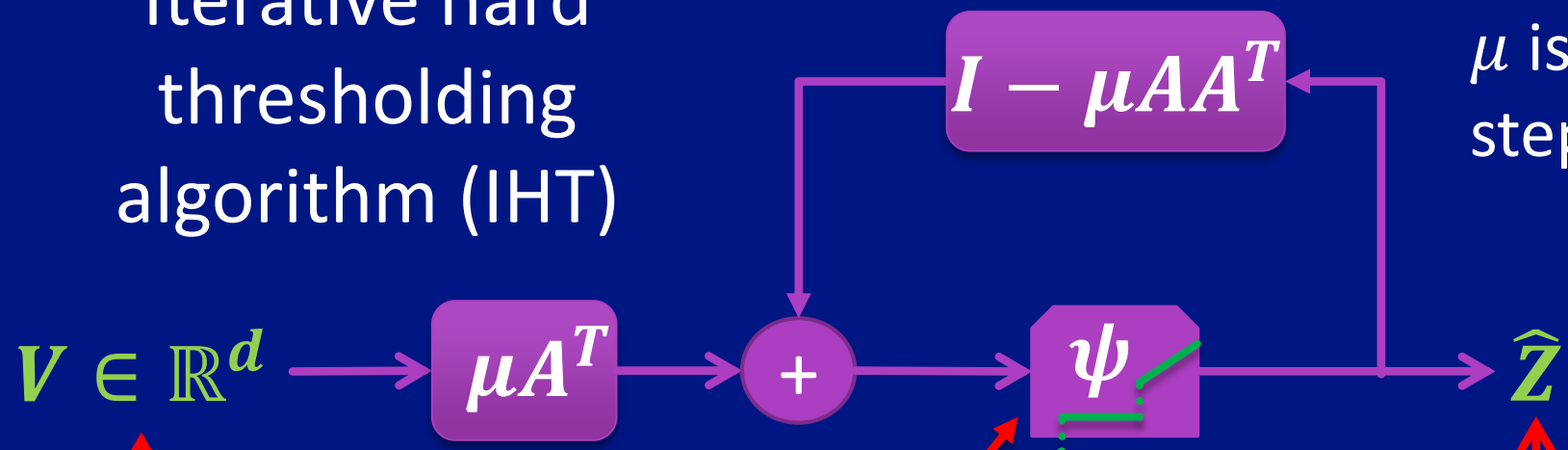
$Z \in \mathcal{Y}$

ψ is a non-linear operation

An estimate of Z

ℓ_0-MINIMIZATION

Iterative hard thresholding algorithm (IHT)



μ is the step size

$V = ZA + E$

Z is a k–sparse vecotr

ψ is the hard thresholding operation: keeps the largest k entries

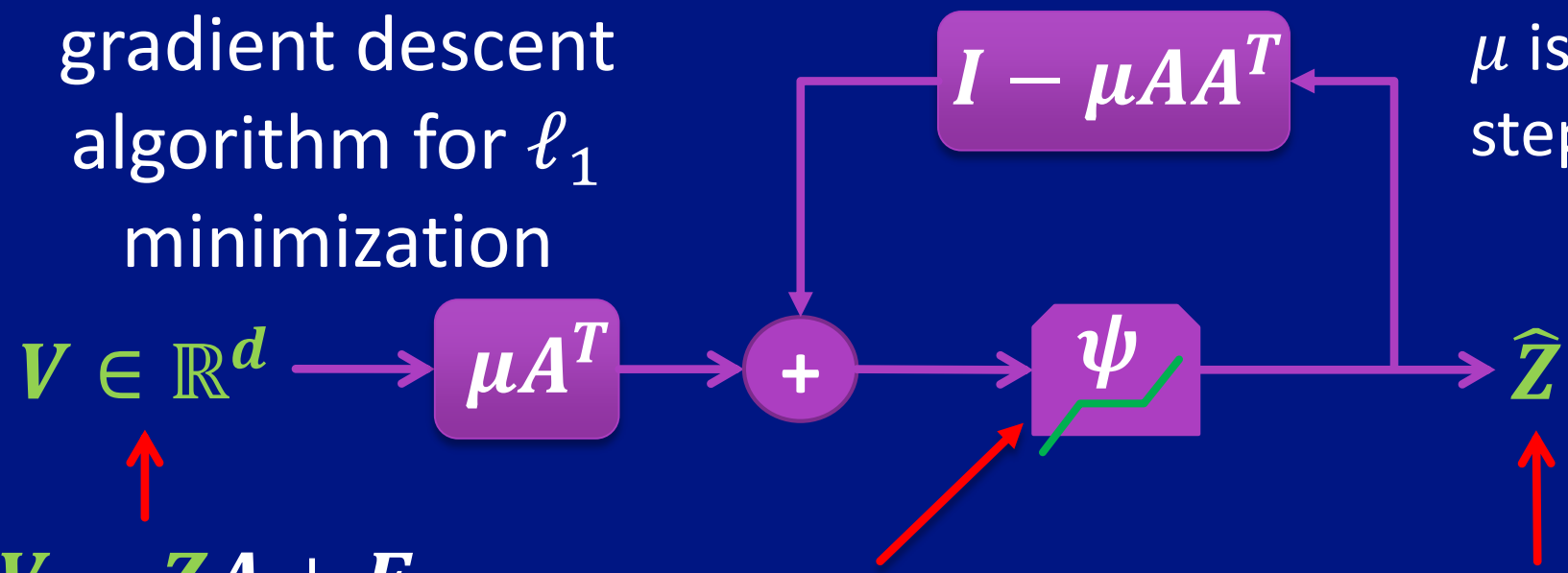
A k-sparse estimate of Z. Aim at solving

$$\min_{\tilde{Z}} \|V - \tilde{Z}A\|$$
$$s.t \ \ \|\tilde{Z}\|_0 \leq \mathrm{k}$$

[Blumensath & Davies, 2009]

ℓ_1-MINIMIZATION

Projected gradient descent algorithm for ℓ_1 minimization

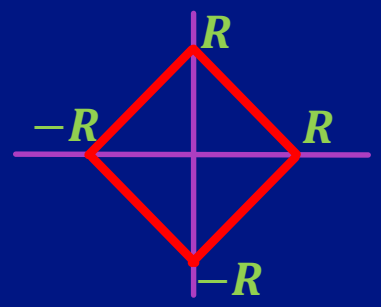


μ is the step size

$V = ZA + E$

$\|Z\|_1 \leq R$

ψ projects onto the ℓ_1 ball



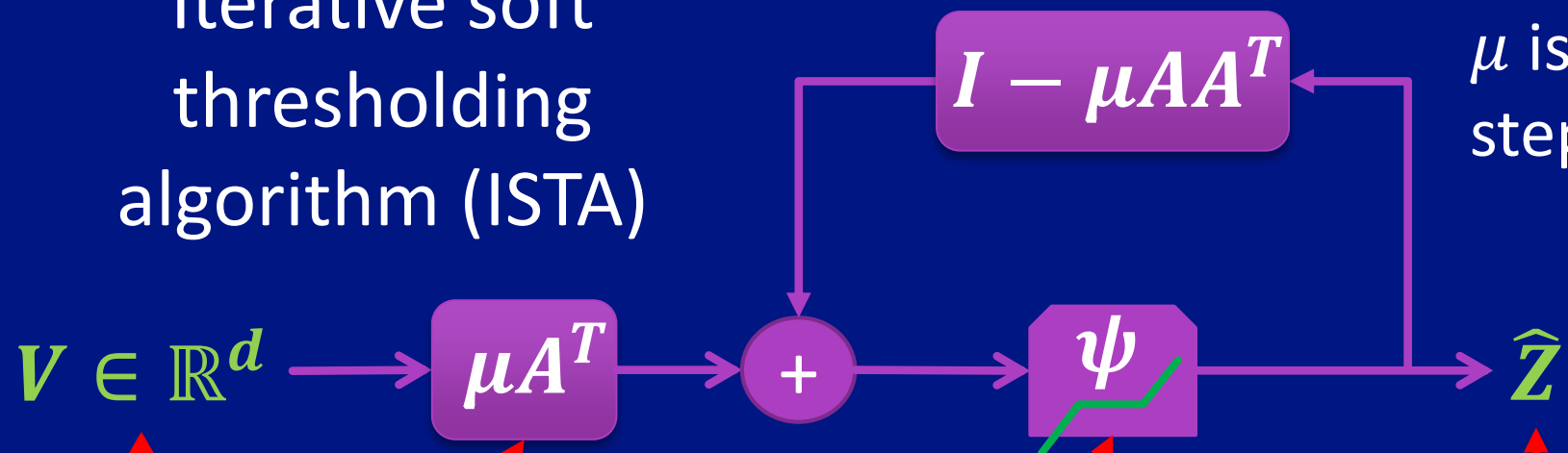
Estimate of Z. Aim at solving

$$\min_{\tilde{Z}} \|V - \tilde{Z}A\|$$
$$s.t \ \|\tilde{Z}\|_1 \leq R$$

UNCONSTRAINED ℓ_1-MINIMIZATION

Iterative soft thresholding algorithm (ISTA)

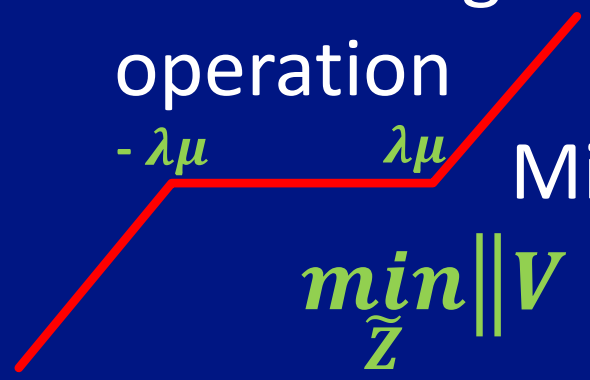
μ is the step size



Step size μ obeys

$$\frac{1}{\mu} \geq \|A\|$$

Soft thresholding operation



Minimizer of

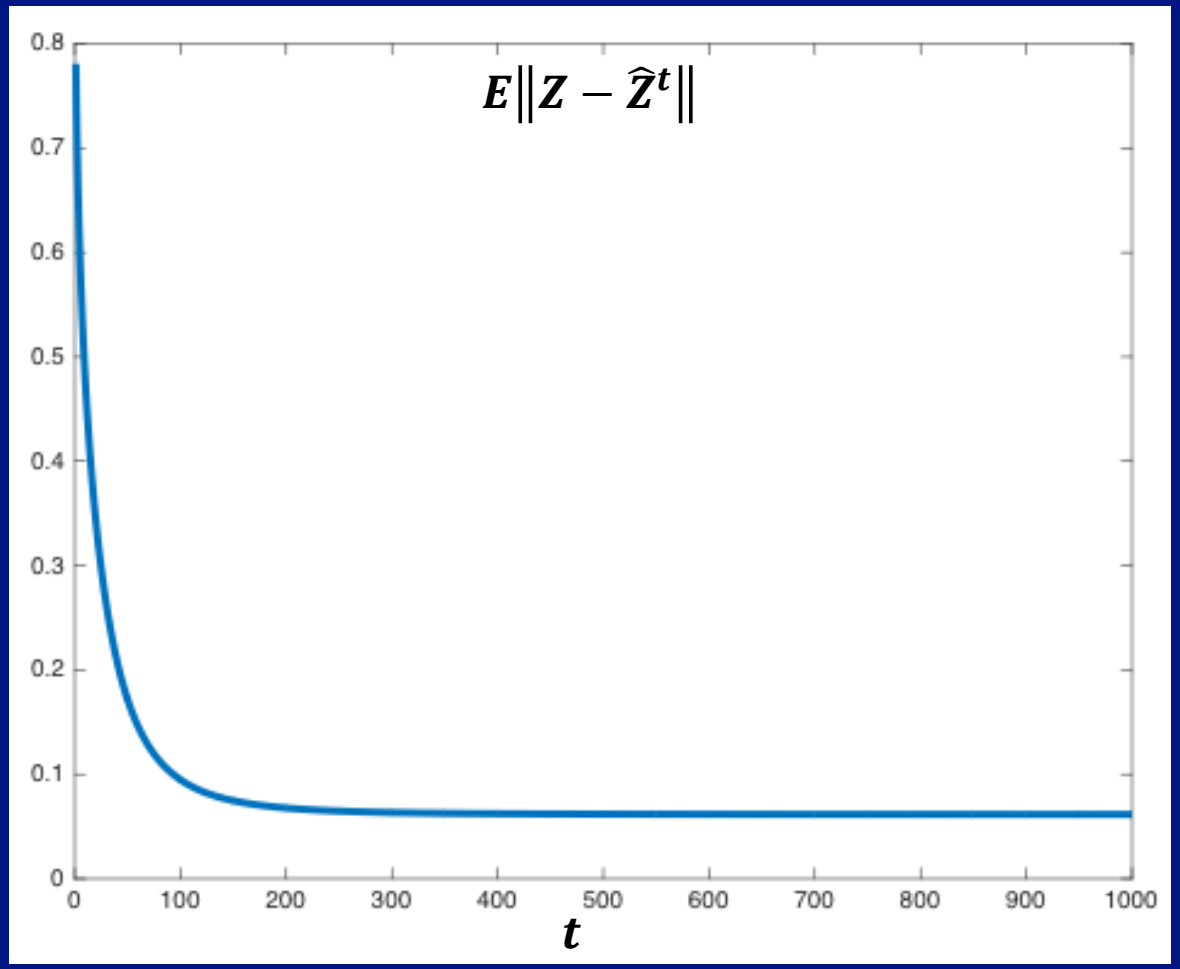
$$\min_{\widetilde{Z}} \|V - \widetilde{Z}A\| + \lambda\|\widetilde{Z}\|_1$$

$$V = ZA + E$$

[Daubechies, Defrise & Mol, 2004], [Beck & Teboulle, 2009]

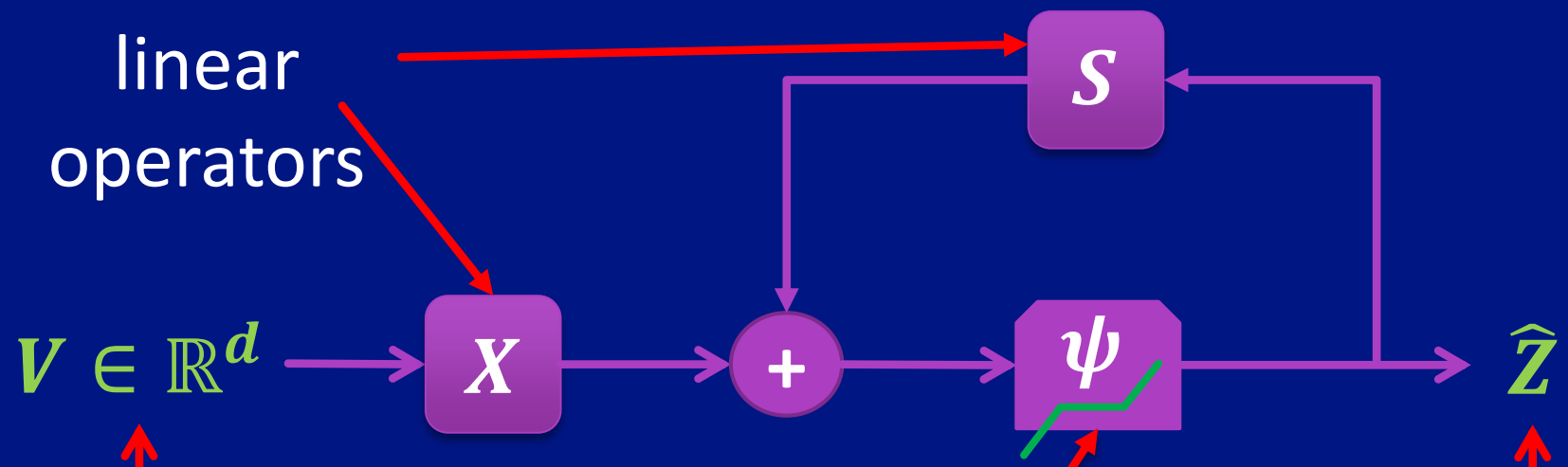
ISTA CONVERGENCE

- Reconstruction mean squared error (MSE) as a function of the number of iterations



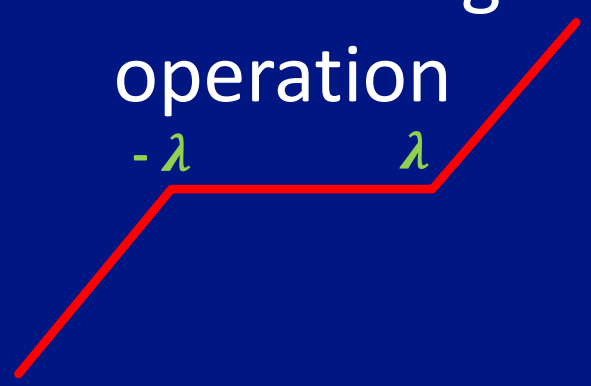
LEARNED ISTA (LISTA)

Learned linear operators



$$V = ZA + E$$

Soft thresholding operation

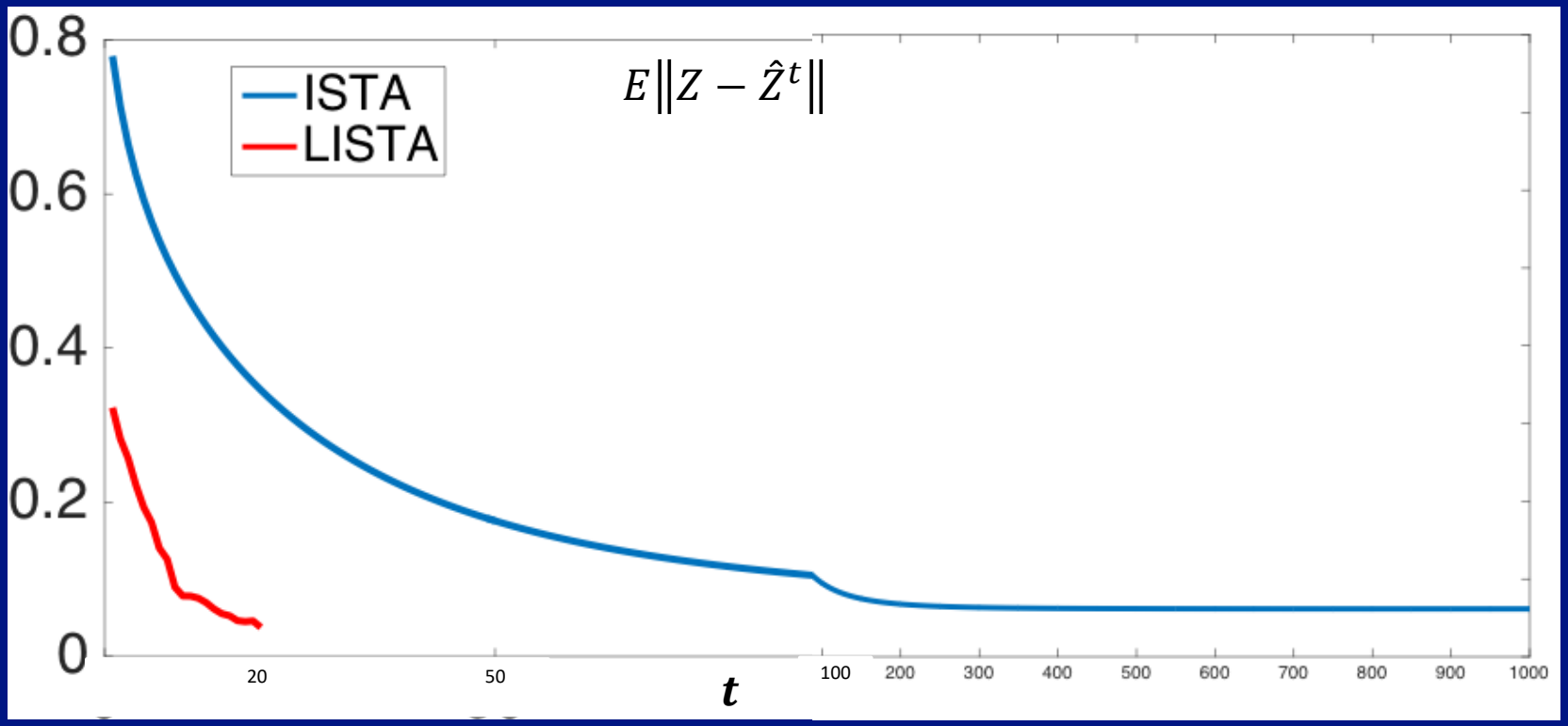


An estimate of Z

[Gregor & LeCun, 2010]

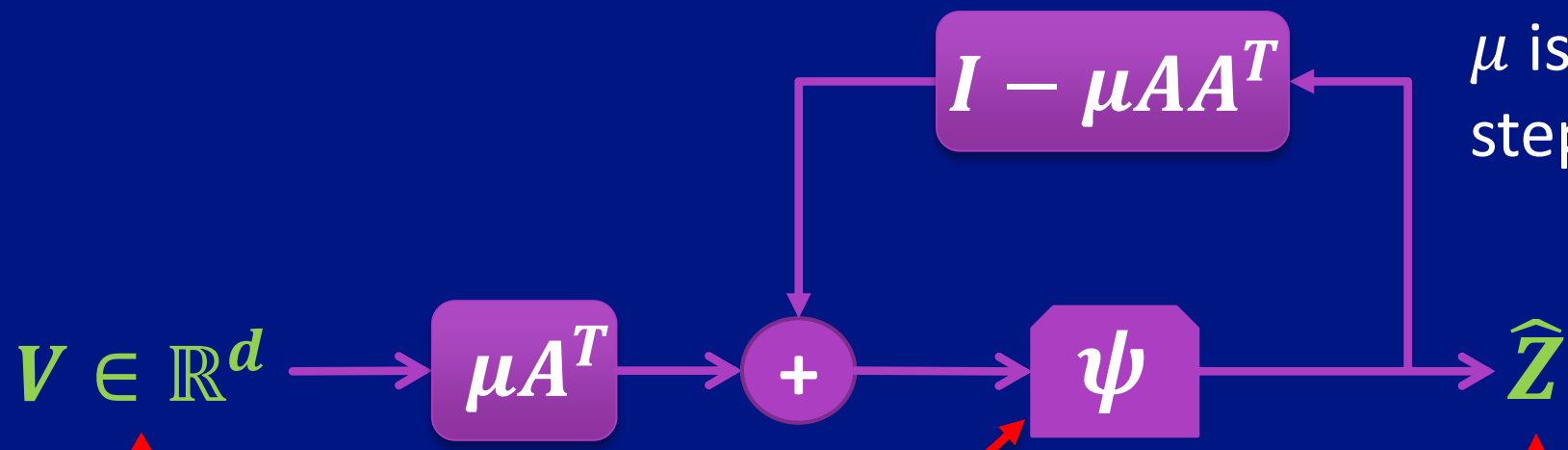
LISTA CONVERGENCE

- Replacing $I - \mu AA^T$ and μA^T in ISTA with the learned X and S improves convergence [Gregor & LeCun, 2010]



- Extensions to other models [Sprechmann, Bronstein & Sapiro, 2015], [Remez, Litani & Bronstein, 2015], [Tompson, Schlachter, Sprechmann & Perlin, 2016].

PROJECTED GRADIENT DESCENT (PGD)

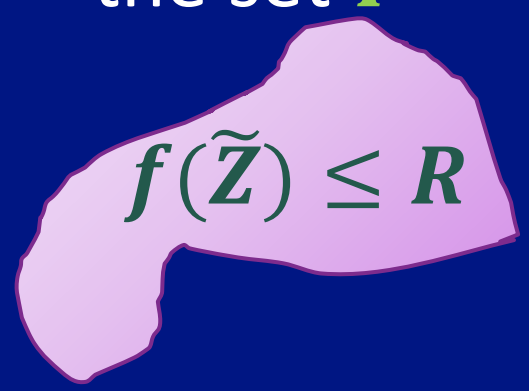


μ is the step size

$V = ZA + E$

$f(Z) \leq R$

ψ projects onto the set $\mathcal{Y}$



Estimate of Z.

Aim at solving

$$\min_{\widetilde{Z}} \|V - \widetilde{Z}A\|$$

$$s.t.\ f(\widetilde{Z}) \leq R$$

THEORY FOR PGD

- Theorem 8: Let $Z \in \mathbb{R}^d$, $f: \mathbb{R}^d \to \mathbb{R}$ a proper function, $f(Z) \leq R$, $C_f(Z)$ the tangent cone of f at point x, $A \in \mathbb{R}^{d \times m}$ a random Gaussian matrix and $V = ZA + E$. Then the estimate of PGD at iteration t, $\hat{Z}^t$, obeys

$$\left\|\hat{Z}^t - Z\right\| \leq \left(\kappa_f \rho\right)^t \|Z\|,$$

where $\rho = \sup_{U,W \in C_f(Z) \cap \mathcal{B}^d} U(I - \mu A A^T)W^T$

and $\kappa_f = 1$ if f is convex and $\kappa_f = 2$ otherwise. [Oymak, Recht & Soltanolkotabi, 2016].

PGD CONVERGENCE RATE

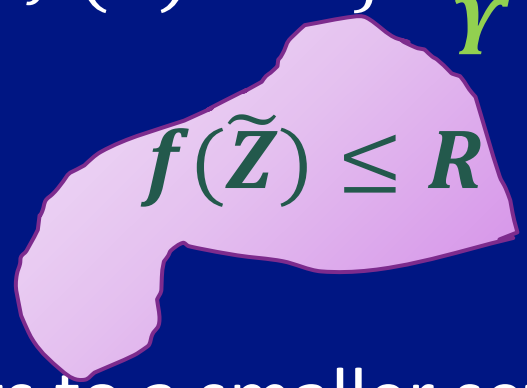
- $\rho = \sup_{U,W \in C_f(Z) \cap \mathcal{B}^d} U(I - \mu AA^T)W^T$ is the convergence rate of PGD.
- Let ω be the Gaussian mean width of $C_f(Z) \cap \mathcal{B}^d$.
- If $\mu = \frac{1}{\left(\sqrt{m}+\sqrt{d}\right)^2} \simeq \frac{1}{d}$ then $\rho = 1 - O\left(\frac{\sqrt{m}-\omega}{m+d}\right)$.
- If $\mu = \frac{1}{m}$ then $\rho = O\left(\frac{\omega}{\sqrt{m}}\right)$.
- For the k-sparse model $\omega^2 = O\big(k\log(\mathrm{d})\big)$
- For GMM with k Gaussians $\omega^2 = O(k)$.
- How may we cause ω to become smaller for having a better convergence rate?

INACCURATE PROJECTION

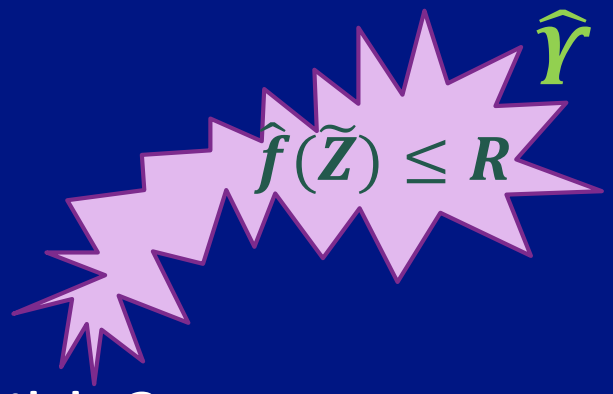
- PGD iterations projects onto $\Upsilon = \{\tilde{Z}: f(\tilde{Z}) \leq R\}$.
- Smaller $\Upsilon \Rightarrow$ Smaller ω.

$\Rightarrow$Faster convergence as

$$\rho = 1 - O\left(\frac{\sqrt{m-\omega}}{m+d}\right) \text{ or } O\left(\frac{\omega}{\sqrt{m}}\right)$$

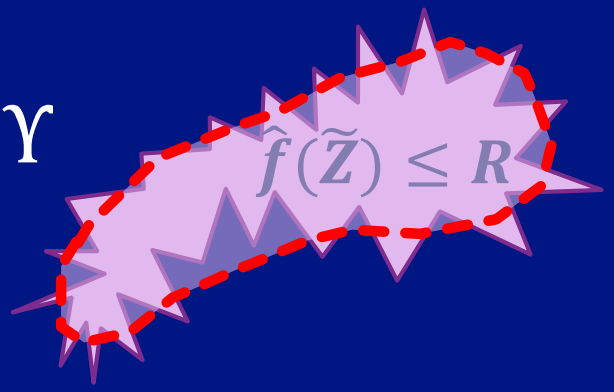


- Let us assume that our signal belongs to a smaller set $\hat{\Upsilon} = \{\tilde{Z}: \hat{f}(\tilde{Z}) \leq R\}$ with $\hat{\omega} \ll \omega$.
- Ideally, we would like to project onto $\hat{\Upsilon}$ instead of Υ.
- This will lead to faster convergence.
- What if such a projection is not feasible?



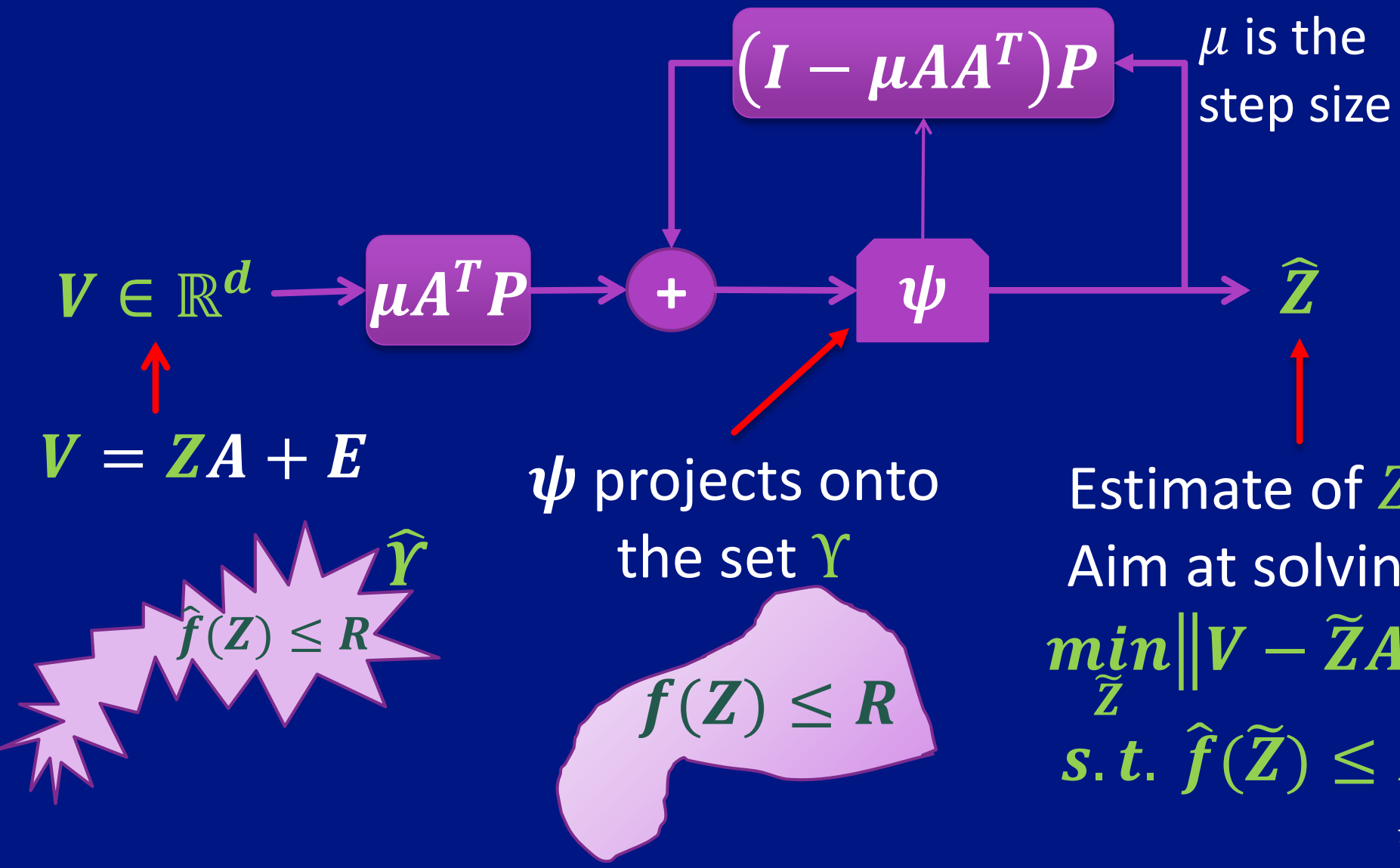
INACCURATE PROJECTION

- We will estimate the projection onto $\widehat{\Upsilon}$ by
  - A linear projection P
  - Followed by a projection onto Υ
- Assumptions:
  - $\|\wp_{\Upsilon}(ZP) - Z\| \leq \epsilon$



Projection of the target vector Z
onto P and then onto Υ

INACCURATE PGD (IPGD)



THEORY FOR IPGD

- Theorem 9: Let $Z \in \mathbb{R}^d$, $f: \mathbb{R}^d \to \mathbb{R}$ a proper convex* function, $f(Z) \leq R$, $\hat{C}_f(Z)$ the tangent cone of f at point Z, $A \in \mathbb{R}^{d \times m}$ a random Gaussian matrix and $V = ZA + E$. Then the estimate of IPGD at iteration t, $\hat{Z}^t$, obeys

$$\left\|\hat{Z}^t - Z\right\| \leq \left((\rho_P)^t + \frac{1 - (\rho_P)^t}{1 - \rho_P} \tilde{\epsilon} \right) \|Z\|,$$

where $\rho_p = \sup_{U,W \in C_f(Z) \cap \mathcal{B}^d} UP(I - \mu AA^T)PW^T$

and $\tilde{\epsilon} = (2 + \rho_p)\epsilon$.

[Giryes, Eldar, Bronstein & Sapiro, 2016]

*We have a version of this theorem also when f is non-proper or non-convex function

CONVERGENCE RATE COMPARISON

- PGD convergence:

$$(\rho)^t$$

- IPGD convergence:

$$(\rho_P)^t + \frac{1-(\rho_P)^t}{1-\rho_P}(2+\rho_p)\epsilon$$

$$\overset{(a)}{\cong} (\rho_P)^t + \epsilon \overset{(b)}{\cong} (\rho_P)^t \overset{(c)}{\ll} (\rho)^t$$

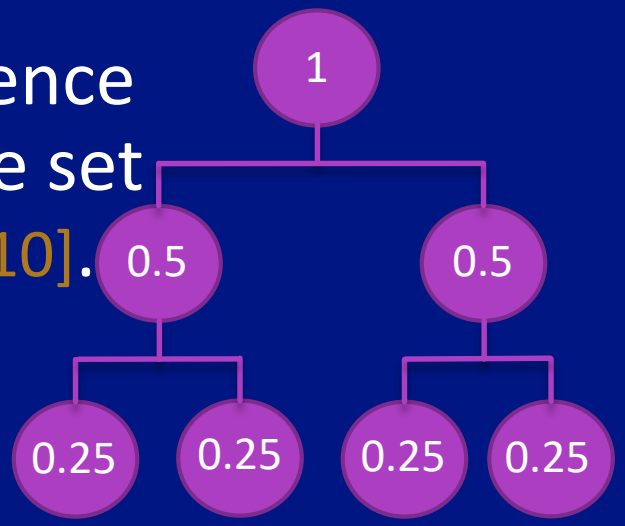
(a) ϵ is negligible compared to ρ_P

(b) For small values of t (early iterations).

(c) Faster convergence as $\rho_P \ll \rho$ (because $\omega_p \ll \omega$).

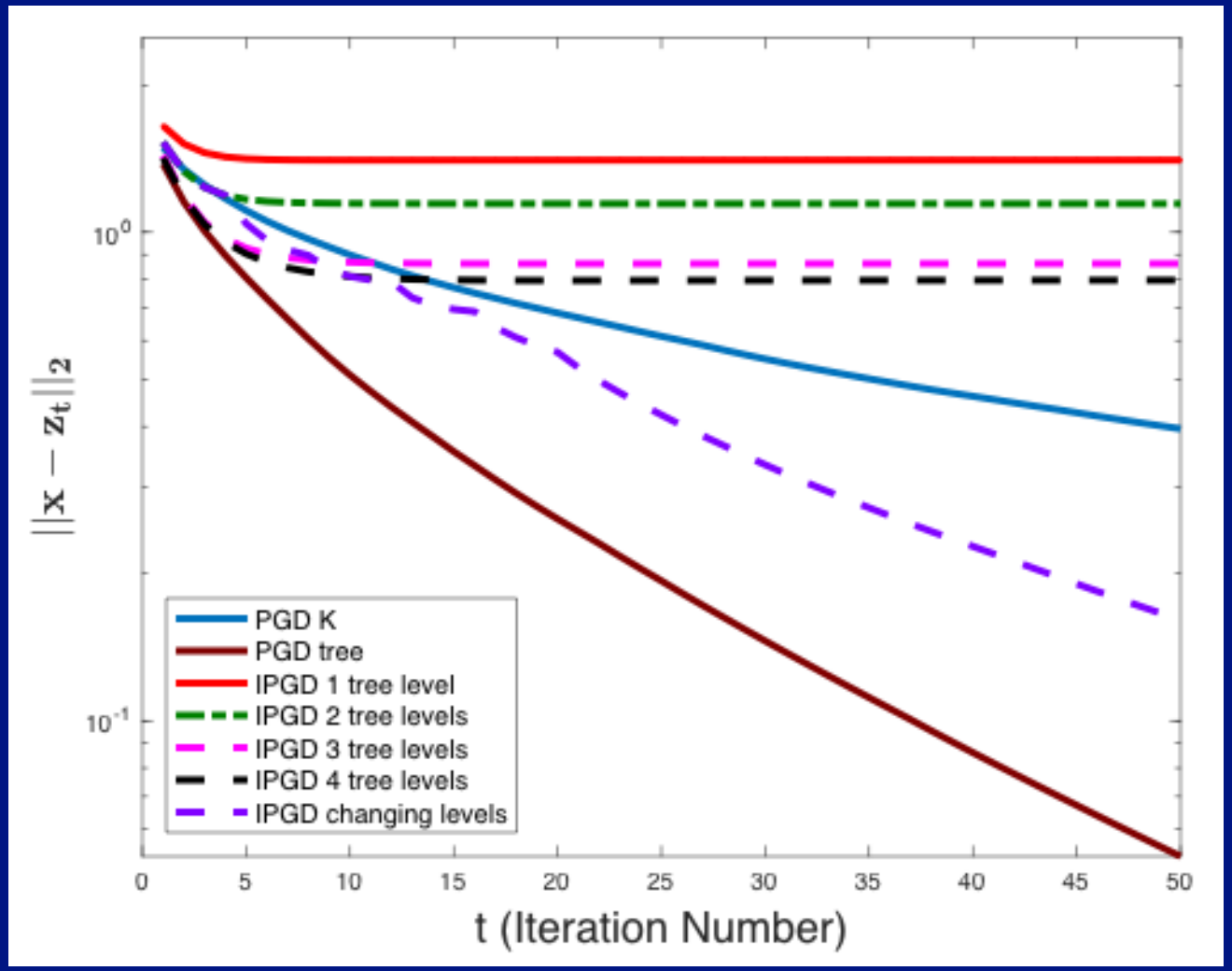
MODEL BASED COMPRESSED SENSING

- $\widehat{\Upsilon}$ is the set of sparse vectors with sparsity patterns that obey a tree structure.
- Projecting onto $\widehat{\Upsilon}$ improves convergence rate compared to projecting onto the set of sparse vectors Υ [Baraniuk et al., 2010].
- The projection onto $\widehat{\Upsilon}$ is more demanding than onto Υ.



- Note that the probability of selecting atoms from lower tree levels is smaller than upper ones.
- P will be a projection onto certain tree levels – zeroing the values at lower levels.

MODEL BASED COMPRESSED SENSING

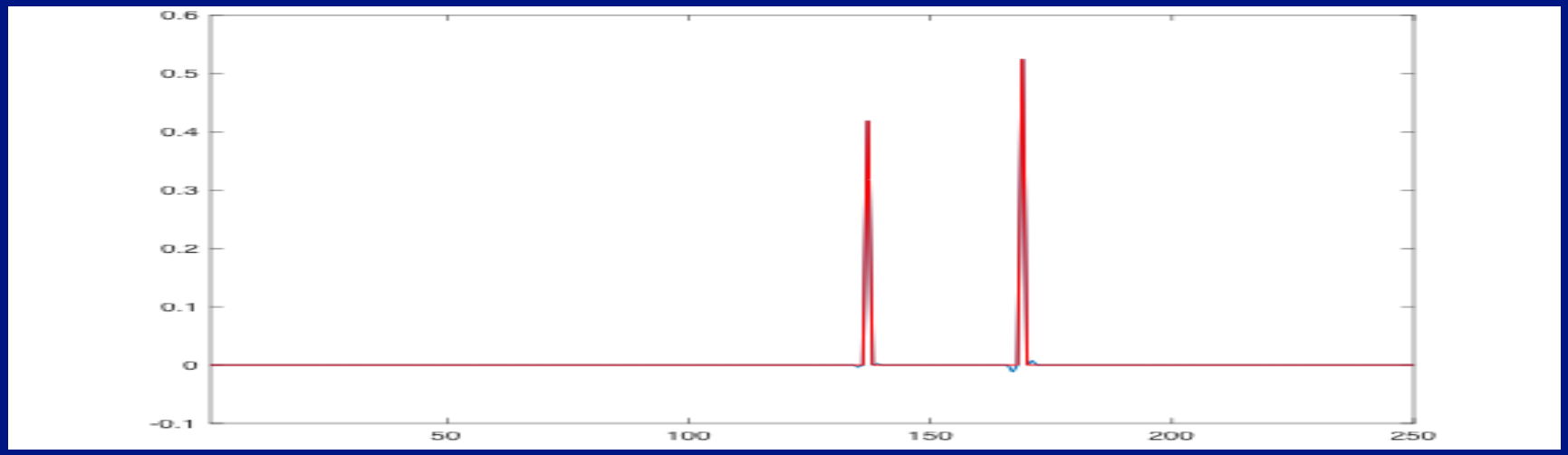


Non-zeros picked entries has zero mean random Gaussian distribution with variance:

- 1 at first two levels
- 0.5^2 at the third level
- 0.2^2 at the rest of the levels

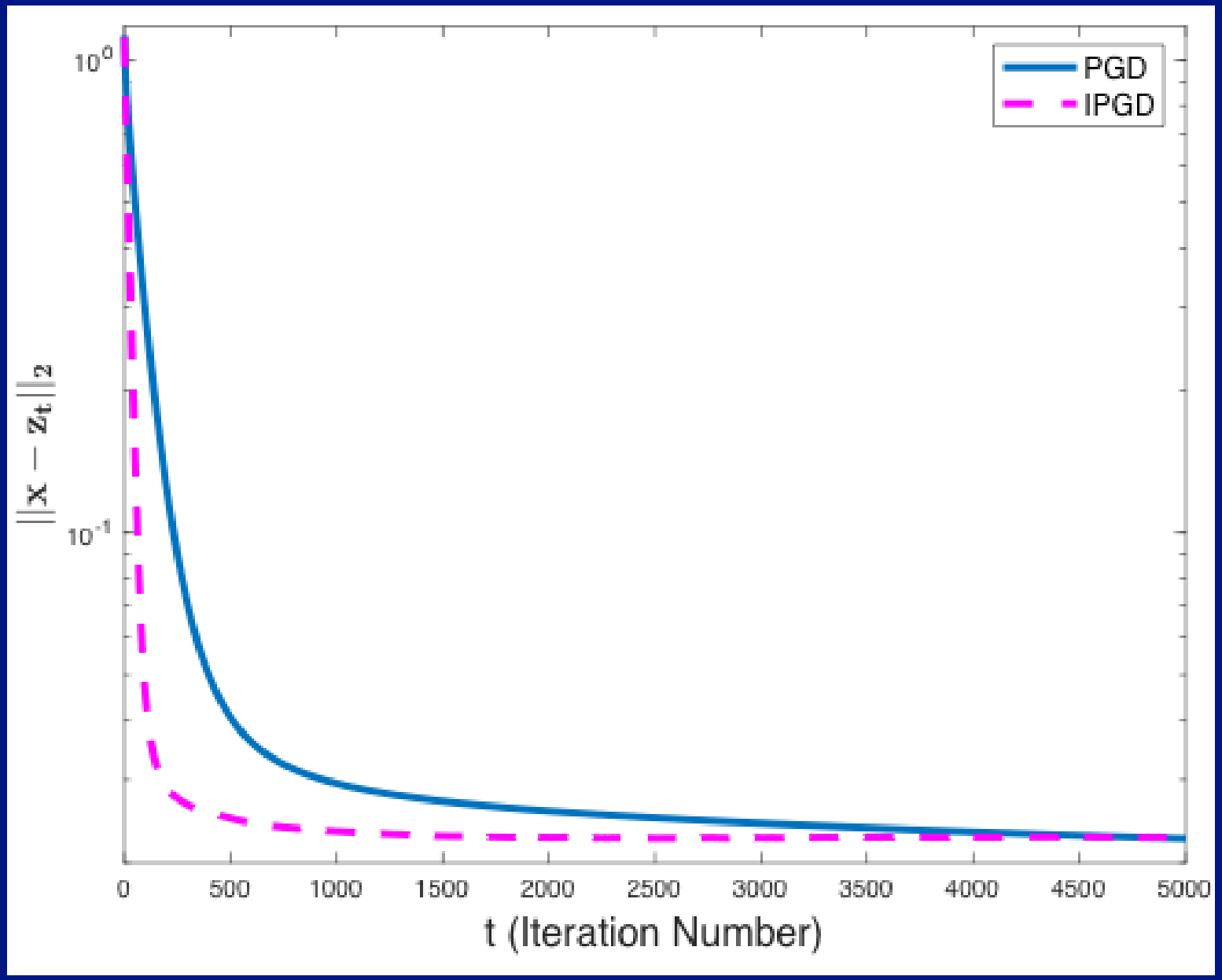
SPECTRAL COMPRESSED SENSING

- $\widehat{\Upsilon}$ is the set of vectors with sparse representation in a 2-times redundant DCT dictionary such that:



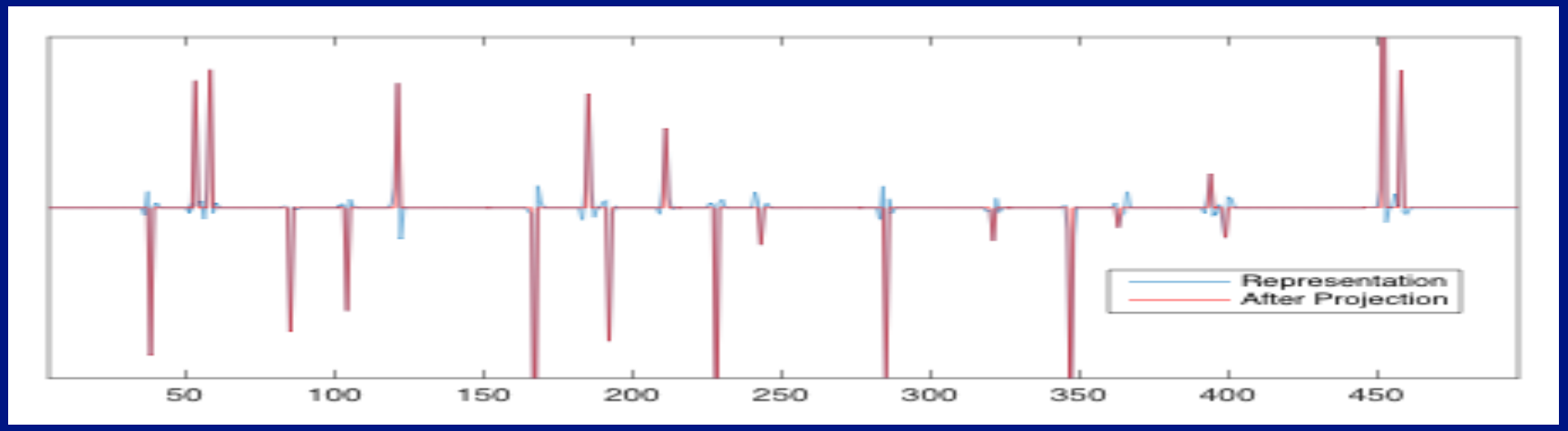
- We set P to be a pooling-like operation that keeps in each window of size 3 only the largest value.

SPECTRAL COMPRESSED SENSING



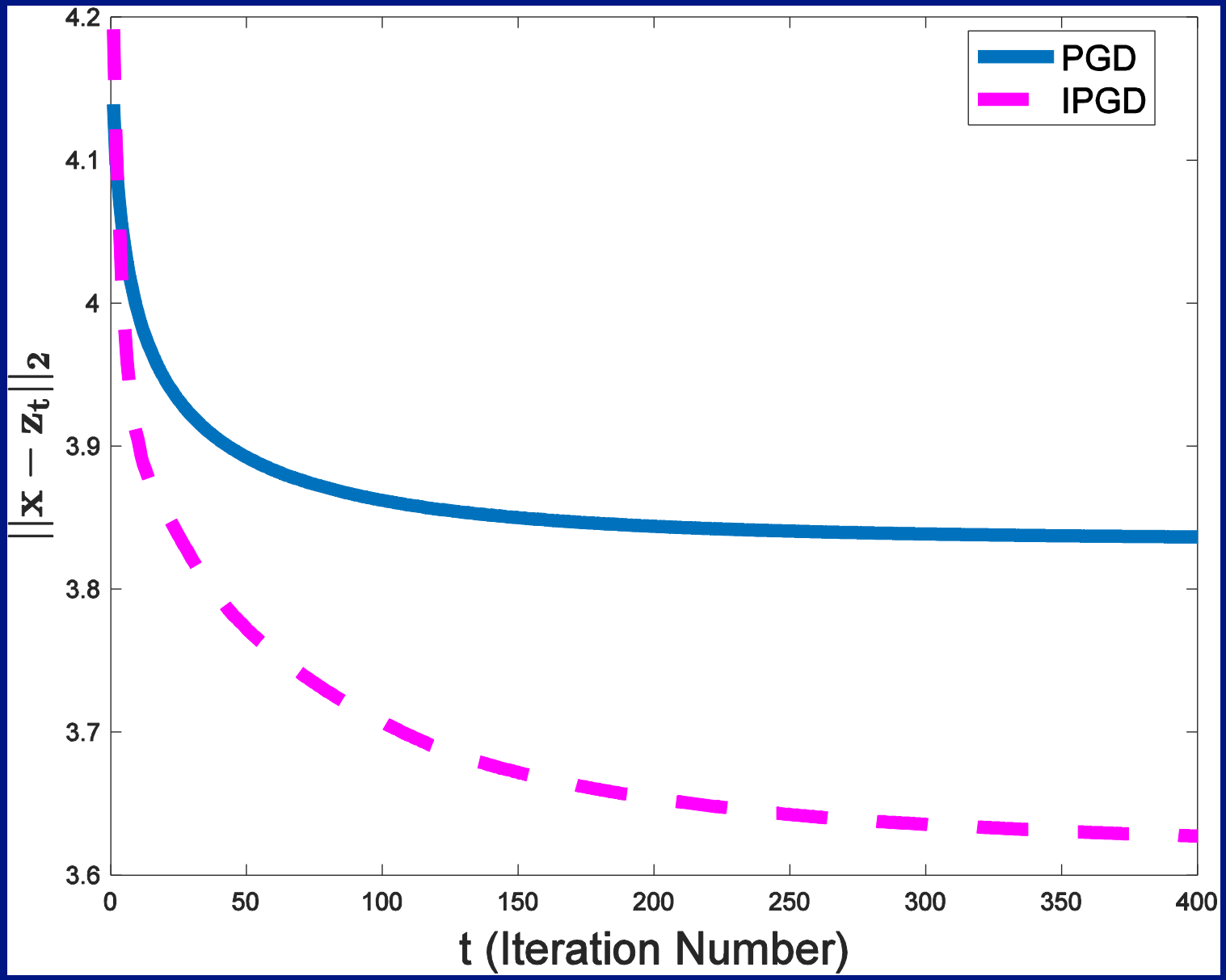
SPECTRAL COMPRESSED SENSING

- $\widehat{\Upsilon}$ is the set of vectors with sparse representation in a 4-times redundant DCT dictionary such that:



- We set P to be a pooling-like operation that keeps in each window of size 5 only the largest value.

SPECTRAL COMPRESSED SENSING

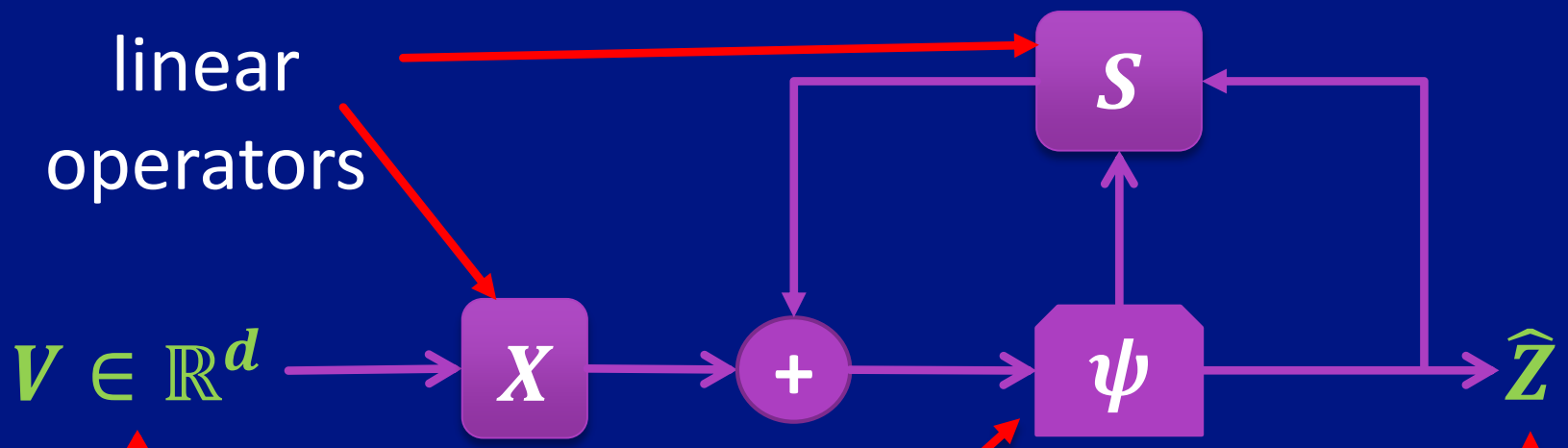


LEARNING THE PROJECTION

- If we have no explicit information about $\widehat{\Upsilon}$ it might be desirable to learn the projection.
- Instead of learning P, it is possible to replace $(I - \mu AA^T)P$ and $\mu A^T P$ with two learned matrices S and X respectively.
- This leads to a very similar scheme to the one of LISTA and provides a theoretical foundation for the success of LISTA.

LEARNED IPGD

Learned linear operators



$V = ZA + E$

$\widehat{\Upsilon}$

$\hat{f}(Z) \leq R$

ψ projects onto the set Υ

$f(Z) \leq R$

Estimate of Z.

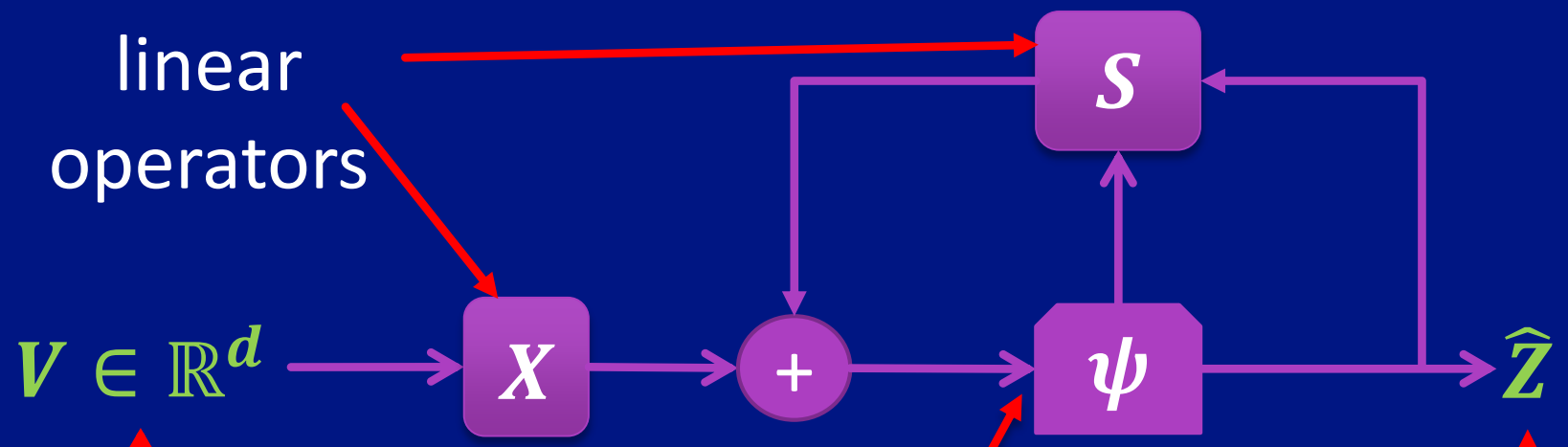
Aim at solving

$$\min_{\widetilde{Z}} \|V - \widetilde{Z}A\|$$

$$s.t.\ \hat{f}(\widetilde{Z}) \leq R$$

LISTA

Learned linear operators



$$V = ZA + E$$

$\widehat{Y}$

$\hat{f}(Z) \leq R$

ψ is a proximal mapping.

$$\psi(U) = \underset{\tilde{Z} \in \mathbb{R}^d}{\operatorname{argmin}} \|U - \tilde{Z}\| + \lambda f(\tilde{Z})$$

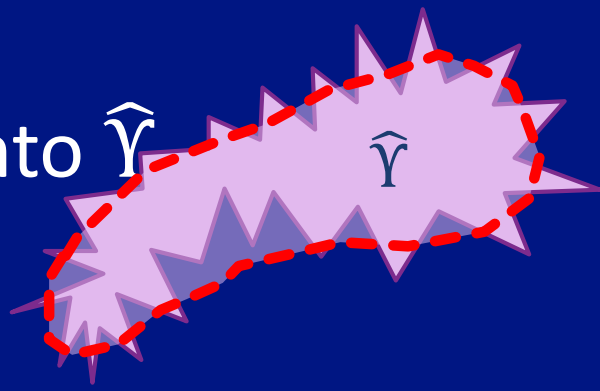
Estimate of Z.

Aim at solving

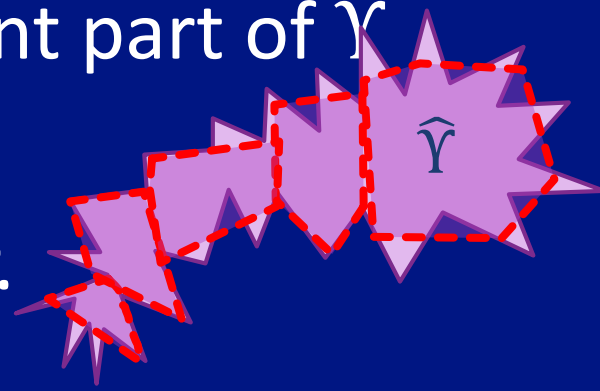
$$\min_{\tilde{Z}} \|V - \tilde{Z}A\| + \lambda \hat{f}(\tilde{Z})$$

LISTA MIXTURE MODEL

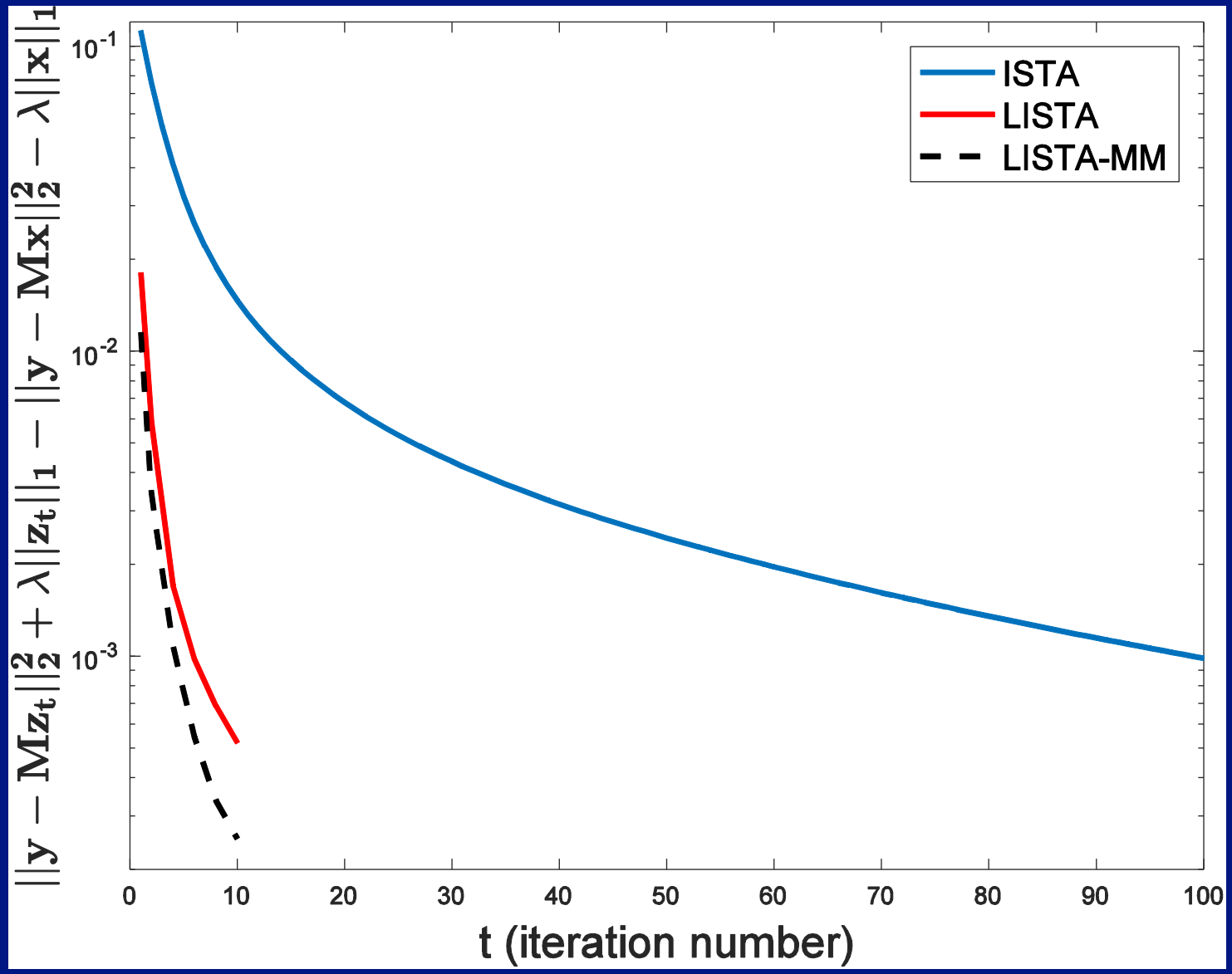
- Approximation of the projection onto $\widehat{\Upsilon}$ with one linear projection may not be accurate enough.



- This requires more LISTA layers/iterations.
- Instead, one may use several LISTA networks, where each approximates a different part of $\widehat{\Upsilon}$
- Training multiple LISTA networks accelerate the convergence further.



LISTA MIXTURE MODEL



RELATED WORKS

- In [Bruna et al. 2017] it is shown that a learning may give a gain due to better preconditioning of *A*.
- In [Xin et al. 2016] a relation to the restricted isometry property (RIP) is drawn
- In [Borgerding & Schniter, 2016] a connection is drawn to approximate message passing (AMP).
- All these works consider only the sparsity case

Take Home Message

DNN keep the important information of the data.

Gaussian mean width is a good measure for the complexity of the data.

Important goal of training: Classify the boundary points between the different classes in the data.

Random Gaussian weights are good for classifying the average points in the data.

DNN may solve optimization problems

Deep learning can be viewed as a metric learning.

Generalization error depends on the DNN input margin

ACKNOWLEDGEMENTS

Yonina C. Eldar
Technion

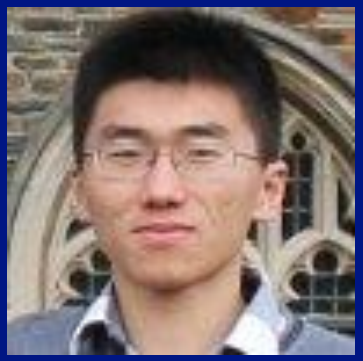
Guillermo Sapiro
Duke University

Robert Calderbank
Duke University

Miguel Rodrigues
UCL

Alex M. Bronstein
Technion

Qiang Qiu
Duke University



Jiaji Huang
Baidu SVAIL

Jure Sokolic
UCL

QUESTIONS?

WEB.ENG.TAU.AC.IL/~RAJA

FULL REFERENCES 1

- A. L. Samuel, “Some studies in machine learning using the game of checkers,” *IBM Journal*, vol. 3, no. 3, pp. 535–554, 1959.
- D. H. Hubel & T. N. Wiesel, “Receptive fields of single neurones in the cat's striate cortex”, J Physiol., vol. 148, no. 3, pp. 574-591, 1959.
- D. H. Hubel & T. N. Wiesel, “Receptive fields, binocular interaction and functional architecture in the cat's visual cortex”, J Physiol., vol. 160, no. 1, pp. 106-154, 1962.
- K. Fukushima, “Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position”, Biological Cybernetics, vol. 36, no. 4, pp. 93-202, 1980.
- Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard & L. D. Jackel, “Backpropagation Applied to Handwritten Zip Code Recognition”, Neural Computation, vol. 1, no. 4, pp. 541-551, 1989.
- Y.LeCun, L. Bottou, Y. Bengio & P. Haffner, “Gradient Based Learning Applied to Document Recognition”, Proceedings of IEEE, vol. 86, no. 11, pp. 2278–2324, 1998.
- C. Farabet, C. Couprie, L. Najman & Y. LeCun, “Learning Hierarchical Features for Scene Labeling,” IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI), vol. 35, no. 8, pp. 1915-1929, Aug. 2013.
- O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, L. Fei-Fei, “ImageNet Large Scale Visual Recognition Challenge”, International Journal of Computer Vision, vol. 115, no. 3, pp. 211-252, 2015
- A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet classification with deep convolutional neural networks”, NIPS, 2012.
- K. He, X. Zhang, S. Ren, and J. Sun. “Deep residual learning for image recognition”, CVPR, 2016.

FULL REFERENCES 2

- M.D. Zeiler & R. Fergus, “Visualizing and Understanding Convolutional Networks”, ECCV, 2014.
- D. Yu & L. Deng, “Automatic Speech Recognition: A Deep Learning Approach”, Springer, 2014.
- J. Bellegarda & C. Monz, “State of the art in statistical methods for language and speech processing,” Computer Speech and Language, vol. 35, pp. 163–184, Jan. 2016.
- C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke & A. Rabinovich, “Going Deeper with Convolutions”, CVPR, 2015.
- V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra & M. Riedmiller, “Playing Atari with Deep Reinforcement Learning”, NIPS deep learning workshop, 2013.
- V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg & D. Hassabis, “Human-level control through deep reinforcement learning”, Nature vol. 518, pp. 529–533, Feb. 2015.
- D. Silver, A. Huang, C. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel & D. Hassabis, “Mastering the Game of Go with Deep Neural Networks and Tree Search”, Nature, vol. 529, pp. 484–489, 2016.
- S. K. Zhou, H. Greenspan, D. Shen, “Deep Learning for Medical Image Analysis”, Academic Press, 2017.
- I. Sutskever, O. Vinyals & Q. Le, “Sequence to Sequence Learning with Neural Networks”, NIPS 2014.
- A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, L. Fei-Fei, “Large-scale Video Classification with Convolutional Neural Networks”, CVPR, 2014.

FULL REFERENCES 3

- F. Schroff, D. Kalenichenko & J. Philbin, “FaceNet: A Unified Embedding for Face Recognition and Clustering”, CVPR, 2015.
- A. Poznanski & L. Wolf, “CNN-N-Gram for Handwriting Word Recognition”, CVPR, 2016.
- R. Socher, A. Perelygin, J. Wu, J. Chuang, C. Manning, A. Ng & C. Potts, “Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank, EMNLP, 2013.
- H. C. Burger, C. J. Schuler & S. Harmeling, Image denoising: Can plain Neural Networks compete with BM3D?, CVPR, 2012.
- J. Kim, J. K. Lee, K. M. Lee, “Accurate Image Super-Resolution Using Very Deep Convolutional Networks”, CVPR, 2016.
- J. Bruna, P. Sprechmann, and Y. LeCun, “Super-Resolution with Deep Convolutional Sufficient Statistics”, ICLR, 2016.
- V. Nair & G. E. Hinton, “Rectified linear units improve restricted Boltzmann machines”, ICML, 2010.
- L. Deng & D. Yu, “Deep Learning: Methods and Applications”, Foundations and Trends in Signal Processing, vol. 7 no. 3-4, pp. 197–387, 2014.
- Y. Bengio, “Learning Deep Architectures for AI”, Foundations and Trends in Machine Learning, vol. 2, no. 1, pp. 1–127, 2009.
- Y. LeCun, Y. Bengio, & G. Hinton. Deep learning. Nature, vol. 521, no. 7553, pp. 436–444, 2015.
- J. Schmidhuber, “Deep learning in neural networks: An overview”, Neural Networks, vol. 61, pp. 85–117, Jan. 2015.
- I. Goodfellow, Y. Bengio & A. Courville, “Deep learning”, Book in preparation for MIT Press, 2016.

FULL REFERENCES 4

- G. Cybenko, “Approximation by superpositions of a sigmoidal function,” Math. Control Signals Systems, vol. 2, pp. 303–314, 1989.
- K. Hornik, “Approximation capabilities of multilayer feedforward networks,” Neural Netw., vol. 4, no. 2, pp. 251–257, 1991.
- A. R. Barron, Approximation and estimation bounds for artificial neural networks, Machine Learning, vol. 14, no. 1, pp. 115–133, Jan. 1994.
- G. F. Montúfar & J. Morton, “When does a mixture of products contain a product of mixtures”, SIAM Journal on Discrete Mathematics (SIDMA), vol. 29, no. 1, pp. 321-347, 2015.
- G. F. Montúfar, R. Pascanu, K. Cho, & Y. Bengio, “On the number of linear regions of deep neural networks,” NIPS, 2014.
- N. Cohen, O. Sharir & A. Shashua, “Deep SimNets,” CVPR, 2016.
- N. Cohen, O. Sharir & A. Shashua, “On the Expressive Power of Deep Learning: A Tensor Analysis,” COLT, 2016.
- N. Cohen & A. Shashua, “Convolutional Rectifier Networks as Generalized Tensor Decompositions,” ICML, 2016
- M. Telgarsky, “Benefits of depth in neural networks,” COLT, 2016.
- R. Eldan and O. Shamir, “The power of depth for feedforward neural networks.,” COLT, 2016.
- N. Cohen and A. Shashua, “Inductive Bias of Deep Convolutional Networks through Pooling Geometry,” arXiv abs/ 1605.06743, 2016.
- J. Bruna, Y. LeCun, & A. Szlam, “Learning stable group invariant representations with convolutional networks,” ICLR, 2013.
- Y-L. Boureau, J. Ponce, Y. LeCun, Theoretical Analysis of Feature Pooling in Visual Recognition, ICML, 2010.

FULL REFERENCES 5

- J. Bruna, A. Szlam, & Y. LeCun, “Signal recovery from lp pooling representations”, ICML, 2014.
- S. Soatto & A. Chiuso, “Visual Representations: Defining properties and deep approximation”, ICLR 2016.
- F. Anselmi, J. Z. Leibo, L. Rosasco, J. Mutch, A. Tacchetti, and T. Poggio, “Unsupervised learning of invariant representations in hierarchical architectures,” Theoretical Computer Science, vol. 663, no. C, pp. 112-121, Jun. 2016.
- J. Bruna and S. Mallat, “Invariant scattering convolution networks,” IEEE Trans. Pattern Analysis and Machine Intelligence (PAMI), vol. 35, no. 8, pp. 1872–1886, Aug 2013.
- T. Wiatowski and H. Bölcskei, “A Mathematical Theory of Deep Convolutional Neural Networks for Feature Extraction,” arXiv abs/1512.06293, 2016
- A. Saxe, J. McClelland, and S. Ganguli, “Exact solutions to the nonlinear dynamics of learning in deep linear neural network”, ICLR, 2014.
- Y. Dauphin, R. Pascanu, C. Gulcehre, K. Cho, S. Ganguli, and Y. Bengio, “Identifying and attacking the saddle point problem in high dimensional non-convex optimization,” NIPS, 2014.
- A. Choromanska, M. B. Henaff, M. Mathieu, G. B. Arous, and Y. LeCun, “The loss surfaces of multilayer networks,” in International Conference on Artificial Intelligence and Statistics (AISTATS), 2015.
- B. D. Haeffele and R. Vidal. Global Optimality in Tensor Factorization, Deep Learning, and Beyond. arXiv, abs/1506.07540, 2015.
- S. Arora, A. Bhaskara, R. Ge, and T. Ma, “Provable bounds for learning some deep representations,” in Int. Conf. on Machine Learning (ICML), 2014, pp. 584–592.

FULL REFERENCES 6

- A. M. Bruckstein, D. L. Donoho, & M. Elad, “From sparse solutions of systems of equations to sparse modeling of signals and images”, SIAM Review, vol. 51, no. 1, pp. 34–81, 2009.
- G. Yu, G. Sapiro & S. Mallat, “Solving inverse problems with piecewise linear estimators: From Gaussian mixture models to structured sparsity”, IEEE Trans. on Image Processing, vol. 21, no. 5, pp. 2481 –2499, May 2012.
- N. Srebro & A. Shraibman, “Rank, trace-norm and max-norm,” COLT, 2005.
- E. Cand`es & B. Recht, “Exact matrix completion via convex optimization,” Foundations of Computational mathematics, vol. 9, no. 6, pp. 717– 772, 2009.
- R. G. Baraniuk, V. Cevher & M. B. Wakin, "Low-Dimensional Models for Dimensionality Reduction and Signal Recovery: A Geometric Perspective," Proceedings of the IEEE, vol. 98, no. 6, pp. 959-971, 2010.
- Y. Plan and R. Vershynin, “Dimension reduction by random hyperplane tessellations,” Discrete and Computational Geometry, vol. 51, no. 2, pp. 438–461, 2014.
- Y. Plan and R. Vershynin, “Robust 1-bit compressed sensing and sparse logistic regression: A convex programming approach,” IEEE Trans. Inf. Theory, vol. 59, no. 1, pp. 482–494, Jan. 2013.
- R. Giryes, G. Sapiro and A.M. Bronstein, “Deep Neural Networks with Random Gaussian Weights: A Universal Classification Strategy? “, IEEE Transactions onSignal Processing, vol. 64, no. 13, pp. 3444-3457, Jul. 2016.
- A. Choromanska, K. Choromanski, M. Bojarski, T. Jebara, S. Kumar, Y. LeCun, “Binary embeddings with structured hashed projections”, ICML, 2016.
- J. Masci, M. M. Bronstein, A. M. Bronstein and J. Schmidhuber, “Multimodal Similarity-Preserving Hashing”, IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI), vol. 36, no. 4, pp. 824-830, April 2014.

FULL REFERENCES 7

- H. Lai, Y. Pan, Y. Liu & S. Yan, “Simultaneous Feature Learning and Hash Coding With Deep Neural Networks”, CVPR, 2015.
- A. Mahendran & A. Vedaldi, “Understanding deep image representations by inverting them,” CVPR, 2015.
- K. Simonyan & A. Zisserman, “Very deep convolutional networks for large-scale image recognition”, ICLR, 2015
- A. Krogh & J. A. Hertz, “A Simple Weight Decay Can Improve Generalization”, NIPS, 1992.
- P. Baldi & P. Sadowski, “Understanding dropout”, NIPS, 2013.
- N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, & R. Salakhutdinov, “Dropout: A simple way to prevent neural networks from overfitting,” Journal of Machine Learning Research, vol. 15, no. 1, pp. 1929–1958, 2014.
- L. Wan, M. Zeiler, S. Zhang, Y. LeCun & R. Fergus, “Regularization of Neural Networks using DropConnect”, ICML, 2013.
- S. Ioffe & C. Szegedy, “Batch normalization: Accelerating deep network training by reducing internal covariate shift,” ICML, 2015.
- M. Hardt, B. Recht & Y. Singer, “Train faster, generalize better: Stability of stochastic gradient descent”, ICML, 2016.
- B. Neyshabur, R. Salakhutdinov & N. Srebro, “Path-SGD: Path-normalized optimization in deep neural networks,” NIPS, 2015.
- S. Rifai, P. Vincent, X. Muller, X. Glorot, & Y. Bengio. “Contractive auto-encoders: explicit invariance during feature extraction,” ICML, 2011.

FULL REFERENCES 8

- T. Salimans & D. Kingma, “Weight Normalization: A Simple Reparameterization to Accelerate Training of Deep Neural Networks”, arXiv abs/1602.07868, 2016.
- S. Sun, W. Chen, L. Wang, & T.-Y. Liu, “Large margin deep neural networks: theory and algorithms”, AAAI, 2016.
- S. Shalev-Shwartz & S. Ben-David. “Understanding machine learning: from theory to algorithms”, Cambridge University Press, 2014.
- P. L. Bartlett & S. Mendelson, “Rademacher and Gaussian complexities: risk bounds and structural results”. The Journal of Machine Learning Research (JMLR), vol 3, pp. 463–482, 2002.
- B. Neyshabur, R. Tomioka, and N. Srebro, “Norm-based capacity control in neural networks,” COLT, 2015.
- J. Sokolic, R. Giryes, G. Sapiro, M. R. D. Rodrigues, “Margin Preservation of Deep Neural Networks”, arXiv, abs/1605.08254, 2016.
- H. Xu and S. Mannor. “Robustness and generalization,” JMLR, vol. 86, no. 3, pp. 391–423, 2012.
- J. Huang, Q. Qiu, G. Sapiro, R. Calderbank, “Discriminative Geometry-Aware Deep Transform”, ICCV 2015
- J. Huang, Q. Qiu, G. Sapiro, R. Calderbank, “Discriminative Robust Transformation Learning”, NIPS 2016.
- T. Blumensath & M.E. Davies, “Iterative hard thresholding for compressed sensing”, Appl. Comput. Harmon. Anal, vol. 27, no. 3, pp. 265 – 274, 2009.
- I. Daubechies, M. Defrise, and C. De Mol, “An iterative thresholding algorithm for linear inverse problems with a sparsity constraint”, Communicationson Pure and Applied Mathematics, vol. 57, no. 11, pp. 1413–1457, 2004.

FULL REFERENCES 9

- A. Beck and M. Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. SIAM J. Img. Sci., 2(1):183–202, Mar. 2009.
- K. Gregor & Y. LeCun, “Learning fast approximations of sparse coding”, ICML, 2010.
- P. Sprechmann, A. M. Bronstein & G. Sapiro, “Learning efficient sparse and low rank models”, IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 37, no. 9, pp. 1821–1833, Sept. 2015.
- T. Remez, O. Litany, & A. M. Bronstein, “A picture is worth a billion bits: Real-time image reconstruction from dense binary pixels”, ICCP, 2015.
- J. Tompson, K. Schlachter, P. Sprechmann & K. Perlin, “Accelerating Eulerian Fluid Simulation With Convolutional Networks”, arXiv, abs/1607.03597, 2016.
- S. Oymak, B. Recht, & M. Soltanolkotabi, “Sharp time–data tradeoffs for linear inverse problems”, arXiv, abs/1507.04793, 2016.
- R. Giryes, Y. C. Eldar, A. M. Bronstein, G. Sapiro, “Tradeoffs between Convergence Speed and Reconstruction Accuracy in Inverse Problems”, arXiv, abs/1605.09232, 2016.
- R.G. Baraniuk, V. Cevher, M.F. Duarte & C. Hegde, “Model-based compressive sensing”, IEEE Trans. Inf. Theory, vol. 56, no. 4, pp. 1982–2001, Apr. 2010.
- M. F. Duarte & R. G. Baraniuk, “Spectral compressive sensing”, Appl. Comput. Harmon. Anal., vol. 35, no. 1, pp. 111 – 129, 2013.
- J. Bruna & T. Moreau, Adaptive Acceleration of Sparse Coding via Matrix Factorization, arXiv abs/1609.00285, 2016.