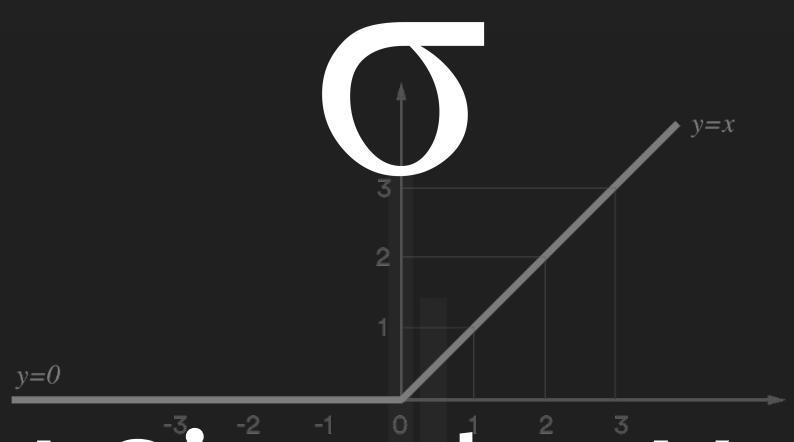


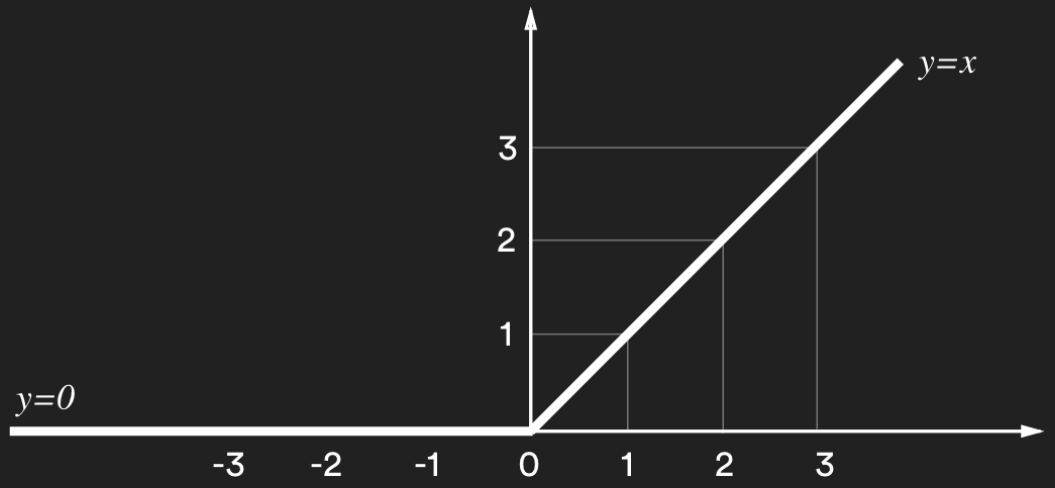
ReLU Singular Values

Sören Dittmer, Emily J. King, Peter Maass

2019

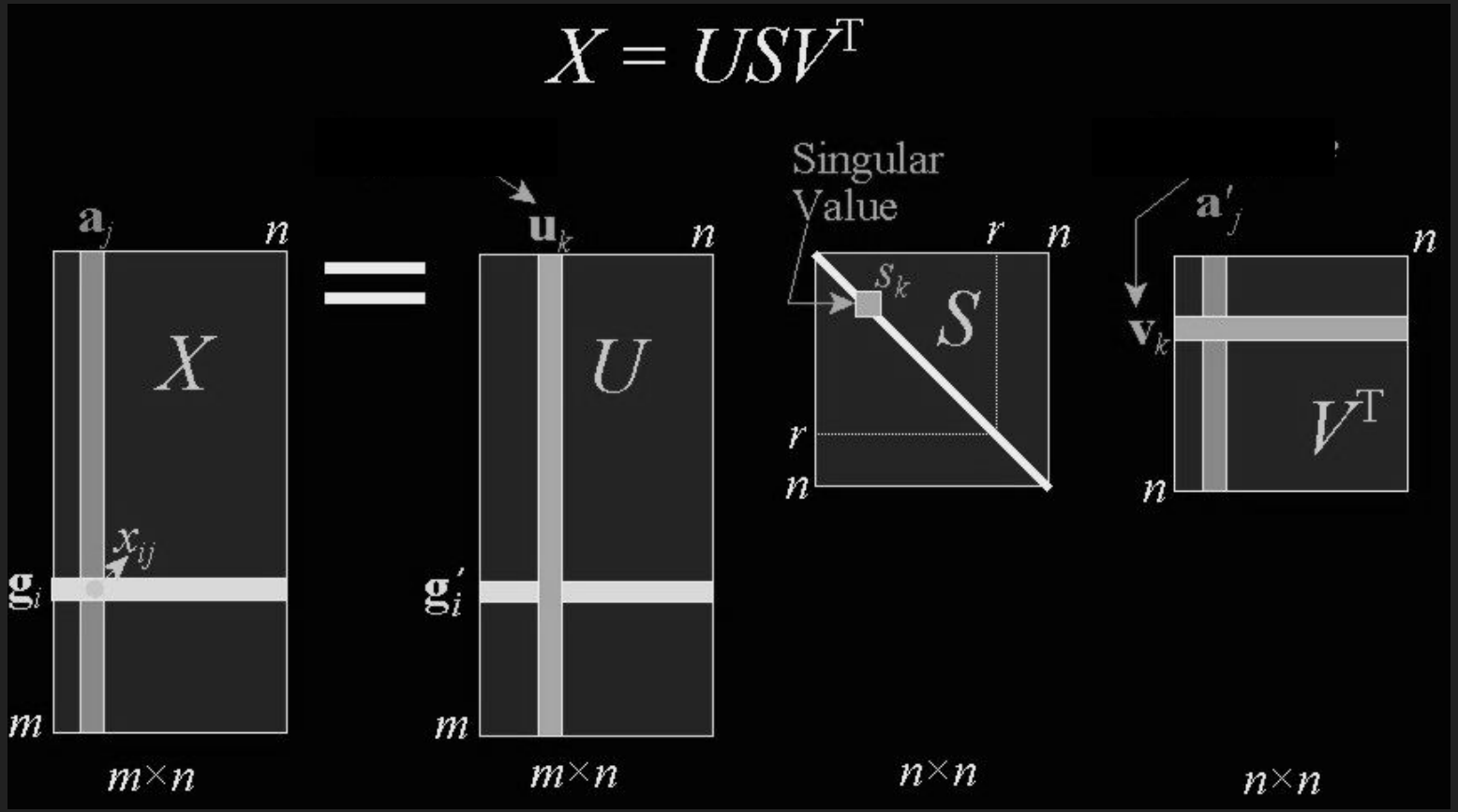
Rectified Linear Units

$$ReLU(x) = \max(x, 0)$$



Introduced in Hahnloser et al. 2000
Used in object recognition in Jarrett et al. 2009
Popularized in RBMs in Nair and Hinton 2010

Singular Value Decomposition



Banach Spaces

A Banach space is a **complete** (cauchy) normed space $(X, ||\cdot||)$, where X is a vector space over a scalar field (such as the real or complex numbers) with a norm $||\cdot||$. This norm induces a metric (the canonical metric) given by

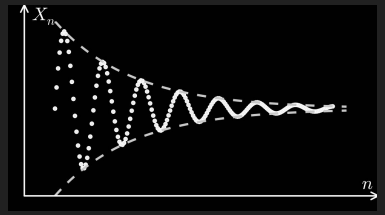
$$d(x, y) := ||y - x|| = ||x - y||$$

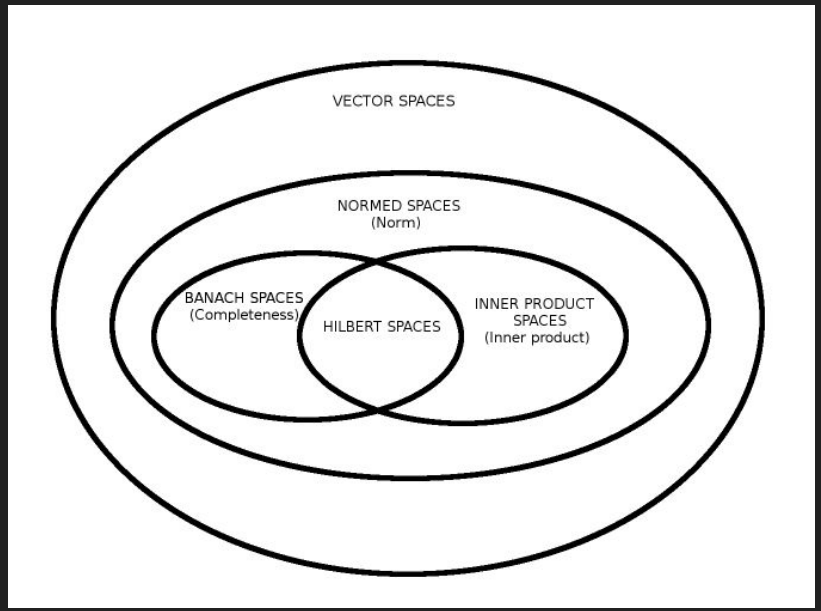
which makes X into a metric space with d.

A sequence $x_\cdot = (x_n)_{n=1}^{\infty}$
is called Cauchy if for every real $r > 0$
There exists an index N such that

$$d(x_n, x_m) = ||x_n - x_m|| < r$$

When n > N, m > N





Operator Norm (General Review)

The operator norm is defined on the space of bounded linear operator between two given normed vector spaces. Informally, the operator norm is a method by which we measure the “size” of a linear operator, or the extent that it can stretch a vector

$$
\begin{aligned}
\|A\|_{\text{op}} &= \inf\{c \geq 0 \ : \ \|Av\| \leq c\|v\| \quad \text{for all} \quad v \in V\} \\
&= \sup\{\|Av\| \ : \ \|v\| \leq 1 \quad \text{and} \quad v \in V\} \\
&= \sup\{\|Av\| \ : \ \|v\| < 1 \quad \text{and} \quad v \in V\} \\
&= \sup\{\|Av\| \ : \ \|v\| \in \{0, 1\} \quad \text{and} \quad v \in V\} \\
&= \sup\{\|Av\| \ : \ \|v\| = 1 \quad \text{and} \quad v \in V\} \quad \text{this equality holds if and only if } V \neq \{ \\
&= \sup\left\{\frac{\|Av\|}{\|v\|} \ : \ v \neq 0 \quad \text{and} \quad v \in V\right\} \quad \text{this equality holds if and only if } V \neq
\end{aligned}
$$

Relationship of Singular Values to Operator Norm

We need a definition that isn't intrinsically tied to linearity...

$$\sigma_k(A) := \min_{L \in \mathbb{R}^{m \times n} : \operatorname{rank} L \leq k} \max_{x \in \mathcal{B}} ||Ax - Lx||_2$$
$$= \min_{L \in \mathbb{R}^{m \times n} : \operatorname{rank}(L) \leq k} ||A - L||_*,$$

The kth singular value of a (linear) operator $A : \mathbb{R}^n \to \mathbb{R}^m$ is the minimal operator norm of $A - L$ with regards to L, where L is a (linear) operator $\mathbb{R}^n \to \mathbb{R}^m$ of rank $\leq k$.

We can also write

$$||A||_* = \max_{x \in \mathcal{B}} ||Ax||_2 = \max_{x \in \mathcal{S}} ||Ax||_2 = \max_{x \in \mathbb{R}^n \setminus \{0\}} \frac{||Ax||_2}{||x||_2},$$

Non-negative Homogeneity (*Definition 1)*

Definition 1: Let V and W be Banach spaces over $\mathbb{R}$. A (possibly nonlinear) operator $\mathcal{A} : V \to W$ is called **nonnegatively homogeneous** if $\mathcal{A}\alpha v = \alpha \mathcal{A} v$ for all $\alpha \geq 0$ and $v \in V$. We then denote the space of all nonnegatively homogeneous operators from V to W as $H_+(V, W)$.

ReLU(x) is non-negative homogenous

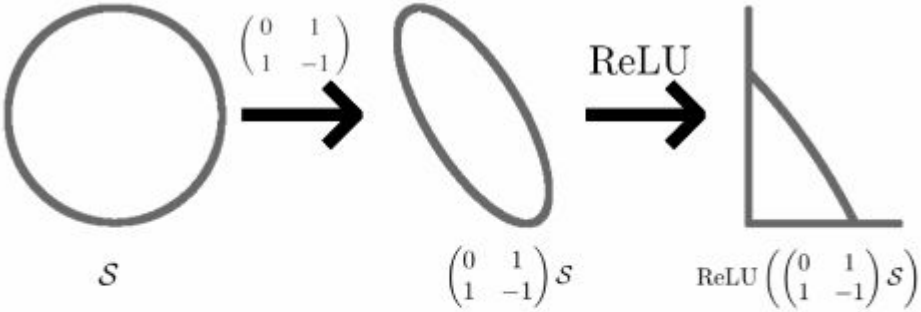


Fig. 1: Example of how the unit sphere gets mapped by a ReLU layer (without bias). Note that the higher the dimensions the layer operates in the more points of $\mathcal{S}$ will be mapped to the sparse "tentacles" on the right.

Operator Norm for Non-Negative Homogenous Operators *(Definition 2)*

Extension of the linear notion of operator norm

Definition 2: Let V and W be Banach spaces over $\mathbb{R}$. The **operator norm** of an $\mathcal{A} \in H_+(V, W)$ is defined as

$$||\mathcal{A}||_* := \sup_{v \in \mathcal{B}} ||\mathcal{A}v||_W, \tag{6}$$

where $\mathcal{B}$ is the unit ball of V.

ReLU Singular Value *(Definition 3)*

This is the principal definition in the paper!

Definition 3: Let $\mathcal{A} \in H_+(\mathbb{R}^n, \mathbb{R}^m)$ be of the form $x \mapsto \mathrm{ReLU}(Ax)$ for some matrix $A \in \mathbb{R}^{m \times n}$. For $k = 0, 1, \ldots, \min\{m, n\} - 1$ we define

$$s_k(\mathcal{A}) = s_k = \min_{\operatorname{rank} L \leq k} \max_{x \in \mathcal{B}} || \mathrm{ReLU}(Ax) - \mathrm{ReLU}(Lx)||_2$$

$$= \min_{\operatorname{rank} L \leq k} \|\mathrm{ReLU}(A\cdot) - \mathrm{ReLU}(L\cdot)\|_* .$$

Like for linear singular values we have the relation

$$s_{k+1}(\mathcal{A}) \leq s_k(\mathcal{A}). \tag{7}$$

Lemma 4

whiteboard

Lemma 4: For $x, y \in \mathbb{R}^n$ we have

$$||\operatorname{ReLU} x - \operatorname{ReLU} y|| \leq ||x - y||.$$

Lemma 5

Lemma 5: Let $\mathcal{A} \in H_+(\mathbb{R}^n, \mathbb{R}^m)$ be given via $\mathcal{A} := \mathrm{ReLU}(A\cdot)$, then

$$s_k(\mathcal{A}) \leq \sigma_k(A).$$

Whiteboard

What does this give us?

Remarks 6,7,8

Remark 6: Although ReLU singular values do not seem to admit a straightforward way to also generalize the singular value decomposition (SVD), they allow for a straightforward definition of a sequence of operators one would associate with the operators given by a truncated SVD in the linear case.

Remark 7: One of the most useful ways to think about a ReLU singular value $s_k(\mathcal{A})$ is to interpret it as the worst case error one has when approximating $\mathcal{A}$ with a rank k approximation.

Remark 8: The concept of ReLU singular values could be easily extended to arbitrary nonnegatively homogeneous operators, e.g. leaky ReLU [21] layers. In fact, one can easily see that Lemmas 4 and 5 still hold when ReLU is replaced with leaky ReLU.

Numerics - Computing Bounds on ReLU Singular Values

notebook

We will now describe a simple numerical method for approximating upper bounds of ReLU singular values that is stronger than Lemma 5. We will then utilize this method to compare these bounds with the singular values of the weight matrices for some random ReLU layers of the form $\mathrm{ReLU}(A\cdot)$. The approximation is a two step process:

1) Approximate

$$W_*, M_* = \underset{\substack{W\in\mathbb{R}^{m\times k}\\ M\in\mathbb{R}^{k\times n}}}{\arg\min} \sum_{x\in X} ||\,\mathrm{ReLU}(Ax) - \mathrm{ReLU}(WMx)||_2^2, \tag{9}$$

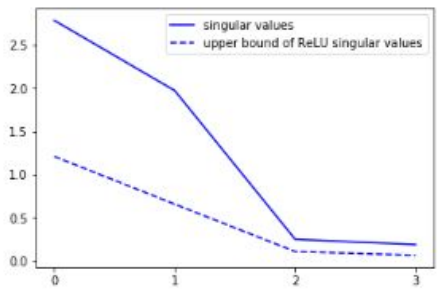
e.g. via the minimization of the function with some kind of stochastic gradient descent (in our case Adam [22]) for X a sufficiently dense finite subset of $\mathcal{S}$.

2) Calculate an approximate upper bound of s_k as

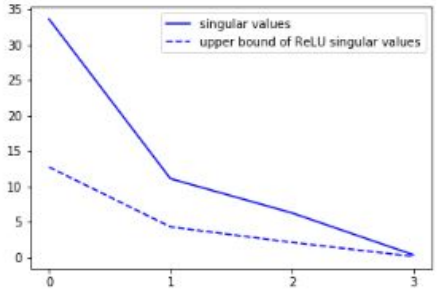
$$\max_{x\in X} ||\,\mathrm{ReLU}(Ax) - \mathrm{ReLU}(W_*M_*x)||_2, \tag{10}$$

this is can be done by simply calculating the norm above for each $x \in X$.

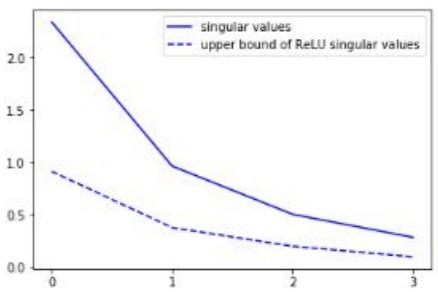
Figure 2



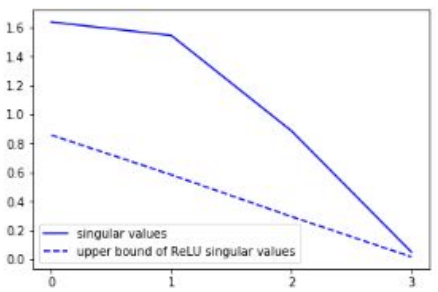
(a) Normal($\mu = 0$, $\sigma^2 = 1$)



(b) Normal($\mu = 0, \sigma^2 = 100$)



(c) Uniform$(0, 1)$



(d) Uniform$(-1, 1)$

Fig. 2: Each plot depicts the singular value curve of a matrix $A \in \mathbb{R}^{4\times4}$ and the corresponding upper bounds of the ReLU singular values of $\mathrm{ReLU}(A\cdot)$ computing via the method outlines in Section II-B. The entries of each respective A are i.i.d. sampled from different distributions for each A respectively. The sampled distributions are labeled beneath the individual plots.

Definition 9 extension to include Biases/Data Dependence

We are interested in how the operator behaves over a subset of the input space, rather than over the entire domain

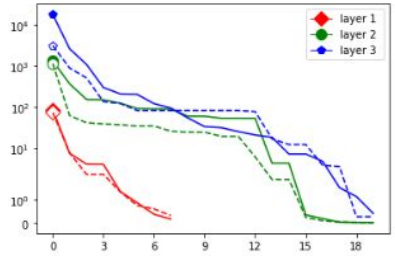
Definition 9: Let $A \in \mathbb{R}^{m \times n}$ and $b \in \mathbb{R}^m$. We then generalize the operator norm to the operator $\mathrm{ReLU}(A \cdot + b)$ over the set $X \subset \mathbb{R}^n$ as

$$||\,\mathrm{ReLU}(A \cdot + b)||_{*,X} = \max_{x \in X} ||\,\mathrm{ReLU}(Ax + b)||_2 \qquad (12)$$

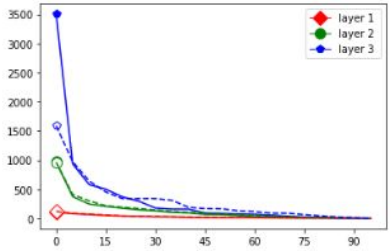
and the kth ReLU singular value of the operator over a given set X as

$$s_{k,X} = \min_{\mathrm{rank}\, L \leq k} \max_{x \in X} ||\,\mathrm{ReLU}(Ax + b) - \mathrm{ReLU}(Lx + b)||_2. \qquad (13)$$

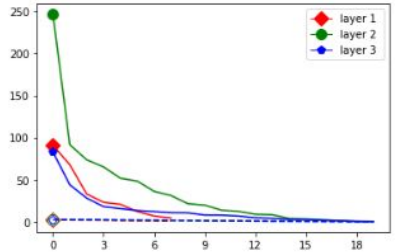
Figure 3



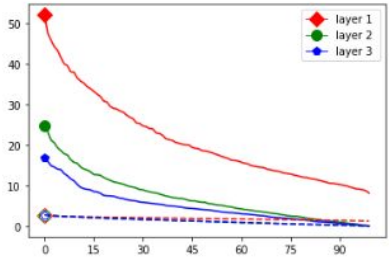
(a) The ReLU layers have a width of 20 and solves the binary classification task giving via the dataset HTRU2 [25]. Note that the vertical axis has a log scale for readability.



(b) The ReLU layers have a width of 100 and the network solves the classical MNIST classification problem [26].



(c) The singular value curve of the weight matrices for of the HTRU2 dataset for comparison. The dashed lines are the singular value curves after the initialization, before the training.



(d) The singular value curve of the weight matrices for of the MNIST dataset for comparison. The dashed lines are the singular value curves after the initialization, before the training.

Fig. 3: The plots (a) and (b) depict a direct comparison of the numerical upper bounds of the data dependent ReLU singular values of the three hidden ReLU layers of a 3 layer MLP. The three ReLU layers are of the same width and the network solves a classification task. The dashed line represents the data-dependent ReLU singular values for misclassified data by the network and the solid line for correctly classified data. The plots (c) and (d) display the singular value curves of the weight matrices of the layers for comparison.

Incorrectly Classified Points

ReLU Layers handle data points that will be classified incorrectly differently than those that are classified.

we briefly want to consider which mechanisms are at play when one applies a layer of the form

$$\mathrm{ReLU}(A\cdot), \quad A \in \mathbb{R}^{m\times n}, \tag{14}$$

to a data point $x \in \mathbb{R}^n$. Using the SVD of $A(= U\Sigma V^T)$ and an on-x-dependent diagonal matrix D_x with 0s and 1s on its diagonal to represent ReLU, we can express the application of this layer to x as

$$\mathrm{ReLU}(Ax) = D_x U\Sigma V^T x. \tag{15}$$

This expression shows that the norm of the output depends on how much x is affected by each singular value of A, i.e., how much it is correlated with which right singular vector and how much ReLU can decrease the impact of the singular value for a given x by setting entries to 0. This can be studied, as Lemma 14 will show, using the Gaussian mean width [7].

Definition 10 - Gaussian Mean Width of a Set

Definition 10: The **Gaussian mean width of a set** $K \subset \mathbb{R}^n$ [7] is defined as

$$\omega(K) := \mathbb{E}_{g \sim \mathcal{N}(0, \mathbb{1}_n)} \sup_{x \in K - K} \langle g, x \rangle , \tag{16}$$

where $\mathcal{N}(0, \mathbb{1}_n)$ denotes an n-dimensional standard Gaussian i.i.d. vector and $K - K$ is to be read in the Minkowski sense, i.e., $K - K = \{x - y : x, y \in K\}$.

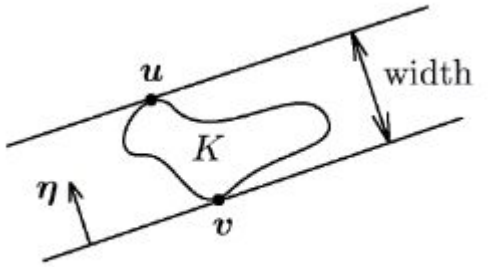


Figure 5: The width of a set in direction η.

Definition - Gaussian Mean width of an Operator

We now introduce the new concept of the **Gaussian mean width of an operator** $\mathcal{A} : \mathbb{R}^n \to \mathbb{R}^m$ as

$$\Omega(\mathcal{A}) := \mathbb{E}_{g \sim \mathcal{N}(0, \mathbb{1}_m)} \sup_{y \in \mathcal{A}(\mathcal{B}) - \mathcal{A}(\mathcal{B})} \langle g, y \rangle , \qquad (17)$$

where $\mathcal{B} \subset \mathbb{R}^n$ is the unit ball. By sampling g from the uniform distribution over the unit sphere instead of from the standard Gaussian, we analogously define $\overline{\omega}$, following [7] and similarly introduce $\overline{\Omega}$ as the **spherical mean width of a set** and **spherical width of an operator**, respectively.

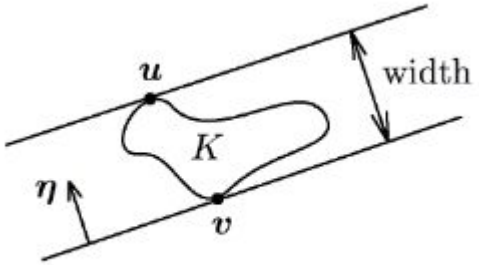


Figure 5: The width of a set in direction η.

Spherical mean width is just the width in all directions

Definition - Spherical Mean Width of Sets and Operators

The *spherical mean width* is then simply the average width, over all directions

$$\tilde{w}(K) = \mathbb{E}\left[\sup_{z \in K-K} \langle \eta, z \rangle\right],$$

where η is distributed uniformly on the sphere.

While the mean width has an intuitive geometric description, the *Gaussian width* is a bit simpler to work with and has similar importance

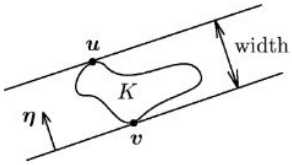


Figure 5: The width of a set in direction η.

Figure 6. Estimating $\boldsymbol{x}$ by blowing up K until it touches the affine subspace $\{\boldsymbol{x}' : \boldsymbol{A}\boldsymbol{x}' = \boldsymbol{y}\}$

Corollary 11

Corollary 11: [7] For all $K \subset \mathbb{R}^n$ we have

$$\omega(\mathrm{Hull}(K)) = \omega(K), \tag{18}$$

where Hull denotes the convex hull.

Corollary 11 shows that – like in the definition of the (linear) operator norm – it does not matter if one defines the operator's Gaussian mean width via the unit ball or the unit sphere.

For more detail [check here]

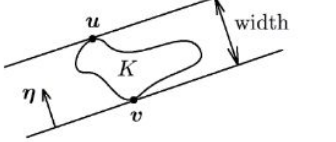


Figure 5: The width of a set in direction η.

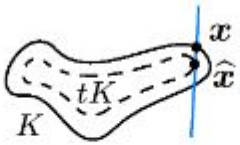
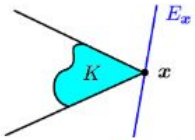
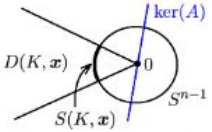


Figure 6. Estimating $\boldsymbol{x}$ by blowing up K until it touches the affine subspace $\{\boldsymbol{x}' : \Lambda \boldsymbol{x}' = \boldsymbol{y}\}$



(a) Descent cone $D(K, x)$ and E_x.



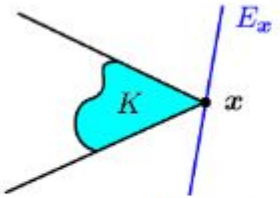
(b) Illustration of recovery condition in terms of the spherical cap $S(K, x)$.

Figure 8: Geometrical concepts involved in exact recovery condition.

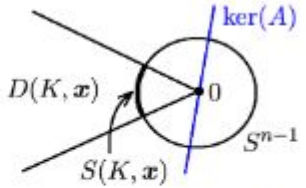
Remark 12

Remark 12: One can estimate a vector $x \in K \subset \mathbb{R}^n$ from m random linear observations where m is proportional to $\omega(K)^2$ with the proportionality factor solely depending on the acceptable (absolute) approximation error. Therefore the Gaussian mean width can be seen as a measure of complexity.

Figure 6. Estimating x by blowing up K until it touches the affine subspace $\{x' : Ax' = y\}$



(a) Descent cone $D(K, x)$ and E_x.



(b) Illustration of recovery condition in terms of the spherical cap $S(K, x)$.

Figure 8: Geometrical concepts involved in exact recovery condition.

Lemmas 13+14

I am going to skip these, but the long and short of it

This lemma demonstrates that the Gaussian mean width of an operator is, at least in the linear case, in some sense a way to measure the accumulative effect of the singular values of an operator and that the Gaussian mean width can also be upper bounded via our more general definition of the operator norm. This makes the Gaussian mean width a good tool for further numerical explorations of the effects seen in Figure 3 and discussed at the beginning of this section.

Numerics

This can be solved by a linear program

Before we can utilize the Gaussian mean width in numerical testing, we derive an algorithm for calculating it. More specifically we want to calculate a good approximation of

$$\omega(K) := \mathbb{E}_{g\sim\mathcal{N}(0,\mathbb{1}_n)} \sup_{x\in K-K} \langle g, x\rangle ,$$

where $K \subset \mathbb{R}^n$ is finite. As argued in [7], due to the Gaussian concentration of measure,

$$\sup_{x\in K-K} \langle g, x\rangle \tag{22}$$

for one $g \sim \mathcal{N}(0, \mathbb{1}_n)$ already yields a good estimate for $\omega(K)$. In practice, to make this estimate more stable, we averaged over the results of 100 samples of g. Due to Corollary 11 we can replace Formula 22 by

$$\sup_{x\in \text{Hull } K-\text{Hull } K} \langle g, x\rangle . \tag{23}$$

element of the set K. This allows us to formulate the solution of Formula 23 via the constraint optimization problem

$$\max_x \langle g, x\rangle , \text{ s.t. } \begin{pmatrix} 1 & \cdots & 1 & 0 & \cdots & 0 \\ 0 & \cdots & 0 & 1 & \cdots & 1 \end{pmatrix} \begin{pmatrix} \alpha \\ .. \\ \beta \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \end{pmatrix}, \tag{24}$$

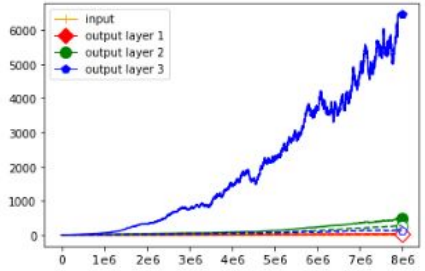
$$(\alpha : \beta) \geq 0 \text{ and} \tag{25}$$

$$(K^T : -K^T)(\alpha : \beta)^T = x, \tag{26}$$

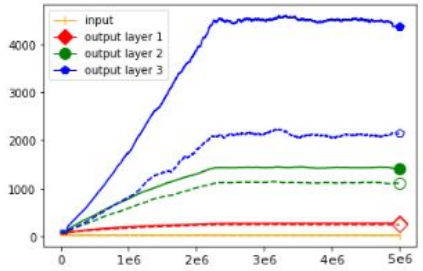
where $\alpha, \beta \in \mathbb{R}^{|K|}$. Rewriting yields the algorithm in form of the following linear program:

$$-\min_{(\alpha:\beta)} \left\langle \begin{pmatrix} -K \\ .. \\ K \end{pmatrix} g, \begin{pmatrix} \alpha \\ .. \\ \beta \end{pmatrix} \right\rangle \text{ s.t. (24), (25) hold.} \tag{27}$$

Figure 4



(a) The Gaussian mean width over the course of the training of the same network used in Figures 3 (a) and (c) (trained on the dataset HTRU2).



(b) The Gaussian mean width over the course of the training of the same network used in Figures 3 (b) and (d) (trained on the dataset MNIST).

Fig. 4: The graphs display how the Gaussian mean width (denoted on the vertical axis) changes on the one hand during the propagation through the layers (different colors and shapes) and on the other hand over the course of the training (horizontal axis) of the networks. For comparability we used the same networks as described in Figure 3. The solid line represents the calculations based on randomly chosen sets of correctly classified data and the dashed line the calculations based on randomly chosen sets of incorrectly classified data of equal size. All graphs where smoothed out for readability. We also checked that the Gaussian mean widths of the input sets (yellow) where on average essentially the same to ensure that we capture differences in processing rather than in data.

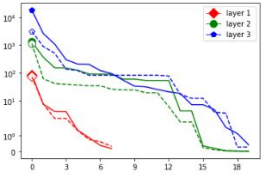
Hypothesis and Summary

From figure 3

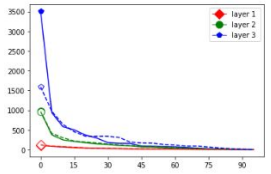
- Some singular values dominate
- Drop off of singular values seen
- What this means:
  - Low rank approximations of a layer work (because the layer is low-rank)
  - Only a few correlations matter!

From fig 4

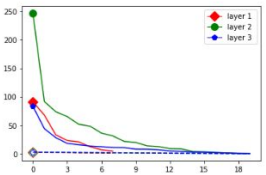
- GMW in network with correct Classification is higher than not
- Misclassification is caused by Lack of correlation with singular Vectors corresponding to "Big" singular values not "blocked" By ReLU



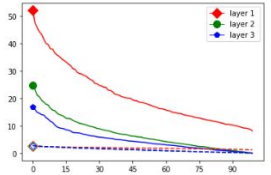
(a) The ReLU layers have a width of 20 and solves the binary classification task giving via the dataset HTRU2 [25]. Note that the vertical axis has a log scale for readability.



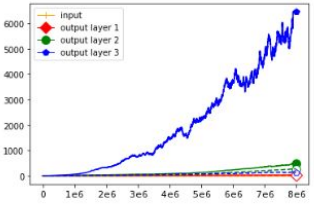
(b) The ReLU layers have a width of 100 and the network solves the classical MNIST classification problem [26].



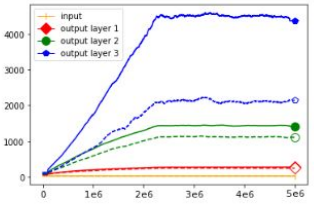
(c) The singular value curve of the weight matrices for of the HTRU2 dataset for comparison. The dashed lines are the singular value curves after the initialization, before the training.



(d) The singular value curve of the weight matrices for of the MNIST dataset for comparison. The dashed lines are the singular value curves after the initialization, before the training.



(a) The Gaussian mean width over the course of the training of the same network used in Figures 3 (a) and (c) (trained on the dataset HTRU2).



(b) The Gaussian mean width over the course of the training of the same network used in Figures 3 (b) and (d) (trained on the dataset MNIST).

Double Layers

layers. We define a (ReLU) **double-layer** as a layer of the form

$$\mathrm{ReLU}(WM \cdot + b), \tag{28}$$

where $W \in \mathbb{R}^{m \times k}$, $M \in \mathbb{R}^{k \times n}$ and $k < \min(m, n)$. This

- We can create and control a linear bottleneck
- These seem to perform better at certain tasks than ReLU
- We can use rank to adjust expressivity
- Can be initialized cleverly (see Definition 15)

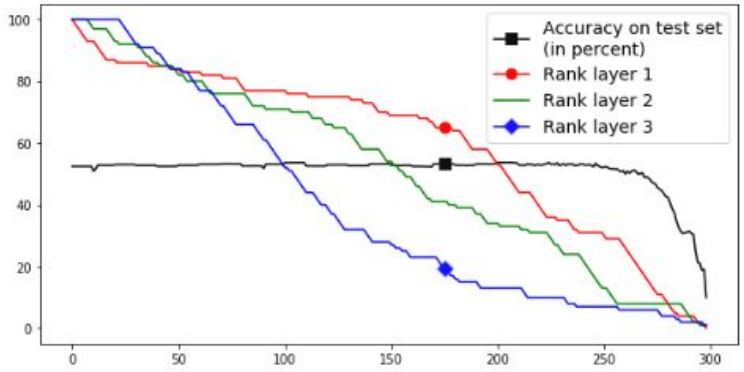
Harmonic Pruning

We can reduce the ranks in layers and reduce number of parameters

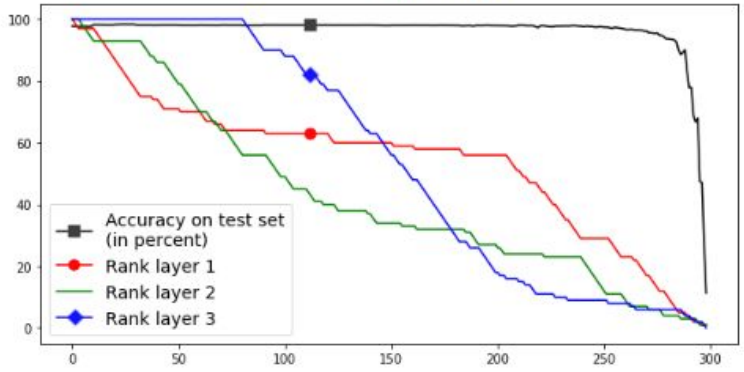
Successively decrease the rank of weight matrices in an already-trained MLP

1) For each layer calculate the change in accuracy of the whole network by setting the smallest non-zero singular value to zero. Denote by A the weight matrix that belongs to the layer that decreases the accuracy the least.
2) Calculate the SVD $A = U\Sigma V^T$ and let $\tilde{\Sigma}$ denote Σ after the smallest non-zero singular value was set to zero. Also define $k := \arg\min_i \left\{\tilde{\Sigma}_{i,i} : \tilde{\Sigma}_{i,i} \neq 0\right\}$ as the index of the smallest non-zero singular value.
3) Reimplement the layer with a split weight matrix replacing A by the rank constrained product WM. Here W is given by the first k columns of $U\sqrt{\tilde{\Sigma}}$ and M is given by the first k rows of $\sqrt{\tilde{\Sigma}}V^T$. W and M are implemented separately to enforce the upper rank (of k) during further steps. Overall we are cutting the rank down by one by setting the smallest non-zero singular value permanently to zero.
4) Retrain the network for some batches if some retraining criterion is if fulfilled, e. g. every 10th iteration or if one of the layers has become very low rank.
5) If some stopping criterion is reached (e. g., the loss

Figure 5



(a) CIFAR-10



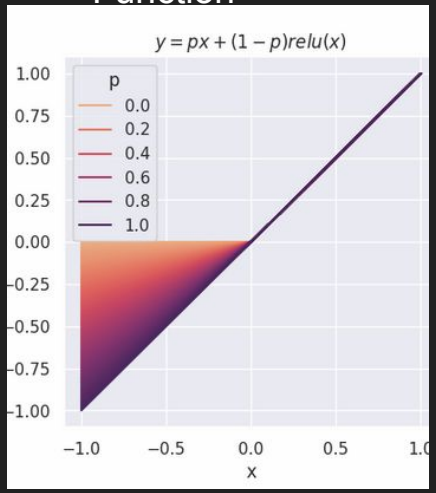
(b) MNIST

Fig. 5: The plot displays the decay of the ranks over the pruning process and the changes in accuracy of the network on the training set. The horizontal axis represents the number of pruning steps, i. e., the iteration of the algorithm.

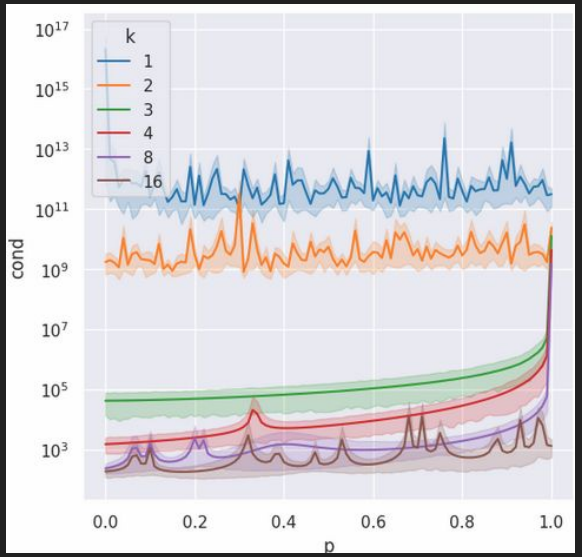
Conclusions and Future Work

- ReLU singular Values and GMW new tools for introspection?
- Or at least some theoretical analysis
- Explain the success of Linear Bottlenecks
- Can these connect to the usual bottlenecks in ResNets?
- Can these measures be useful for matching models and data in transfer learning?
- Can we detect overfitting?

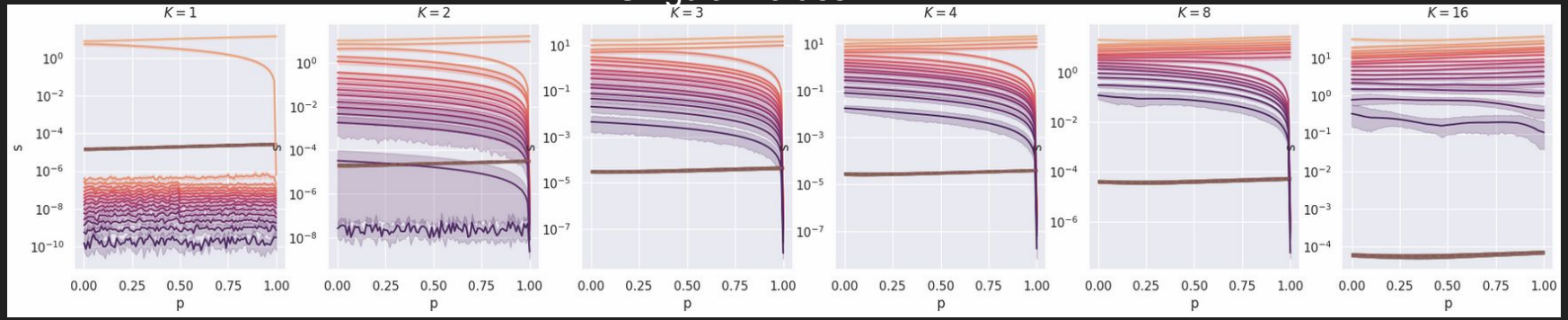
Linear/Nonlinear Function



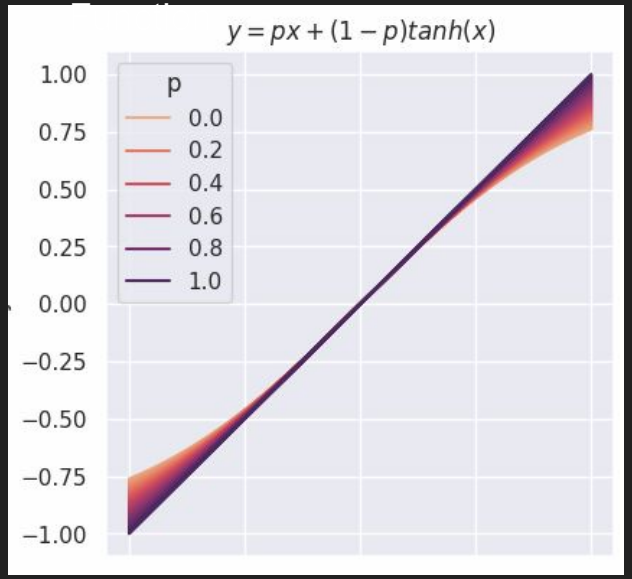
Condition Number



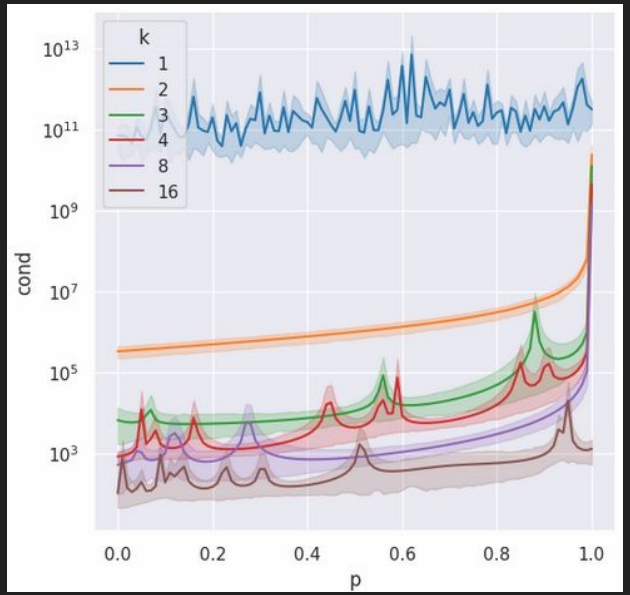
Singular Values



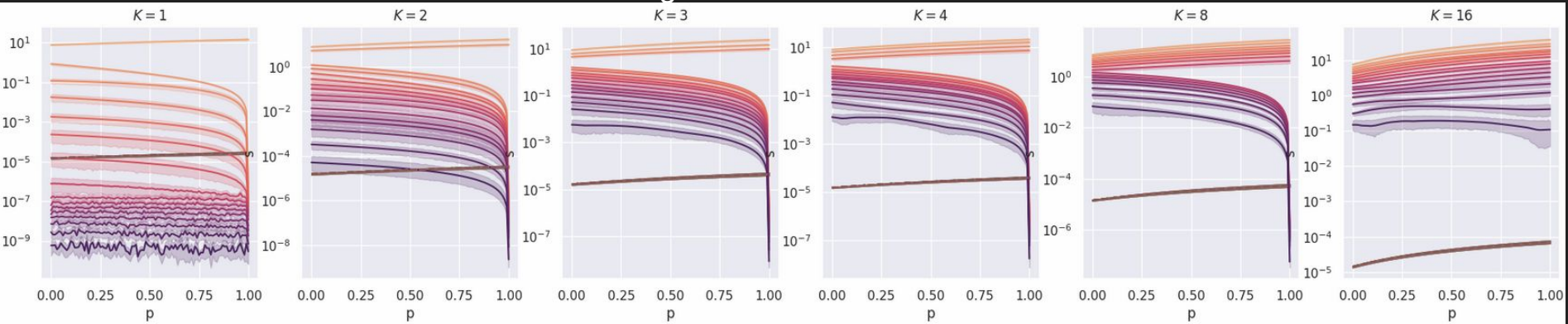
Linear/Nonlinear



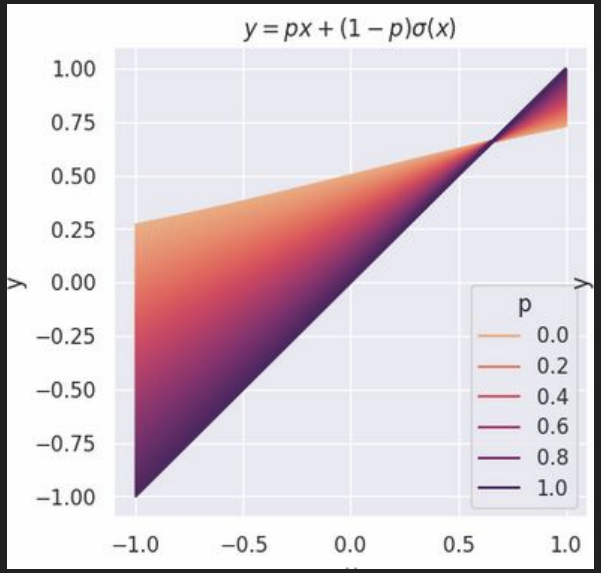
Condition Number



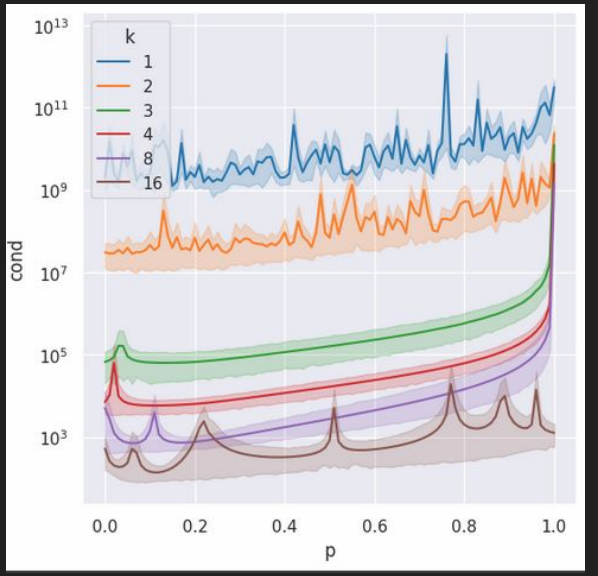
Singular Values



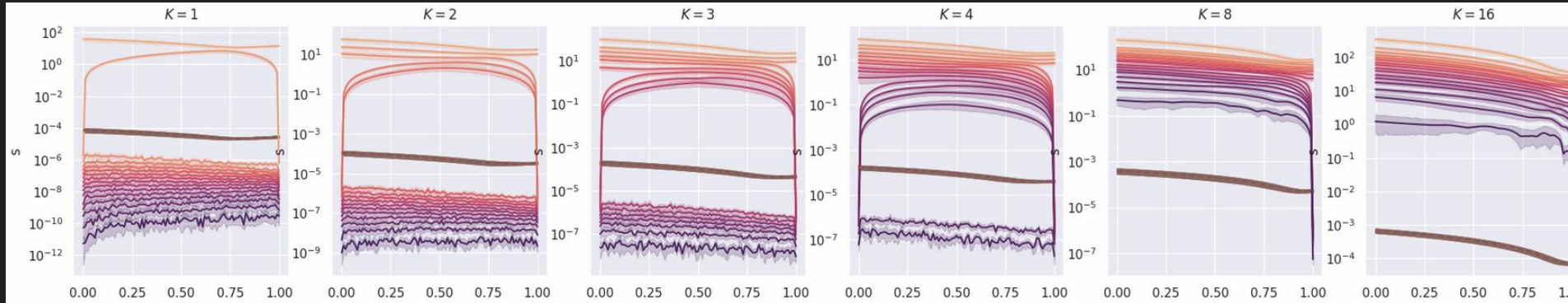
Linear/Nonlinear Function

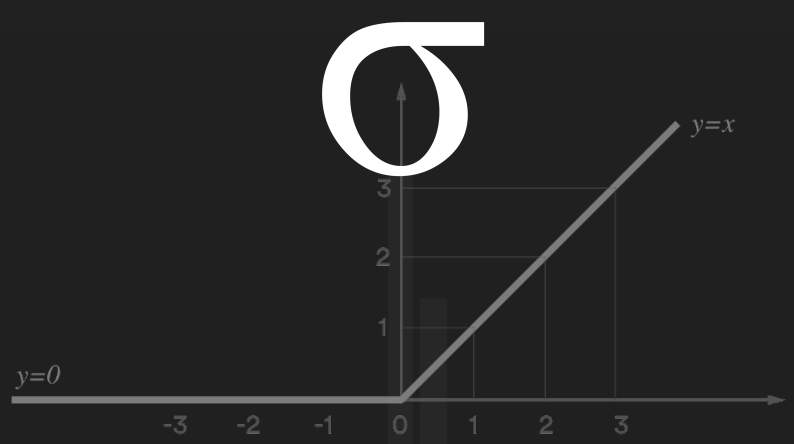


Condition Number



Singular Values





Questions? Thoughts?