Outline for today

- Understanding the filter activations on convolutional nets:

  The first layers are masks by which the image is compared against and responses recorded.

  The activation at a second layer is a linear combination of activations from the first layer and can be viewed as building simple components of an image.

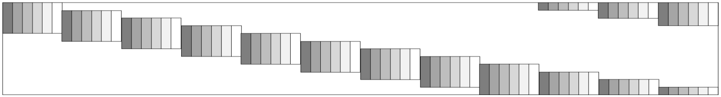
  As depth increases the images that maximize an activation become more like specific images, sometimes referred to as a "grandmother cell".

- The first layer of CNNs are similar to wavelets, which is to be expected and is to be expected due to their near optimality and helps explain transferability

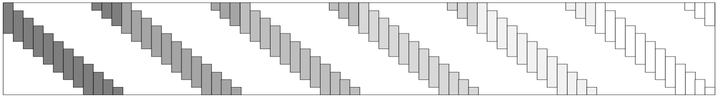
- Focusing on learning one layer, dictionary learning

Convolutional Neural Networks:

Convolutional neural network layers impose a structure on $W^{(i)}$:



(a) A convolutional matrix.



(b) A concatenation of banded and Circulant matrices. [1]

$W^{(i)}$ is composed of a "mask" (usually of compact support, say just living on 9 pixels) translated by some amount which is referred to as "stride." These "masks" are sometimes referred to as "features." Additional nonlinearities include "max pooling."

[1] `https://arxiv.org/pdf/1607.08194.pdf`

Exemplar CNN structure: LeNet5 LeCun et. al. 1998

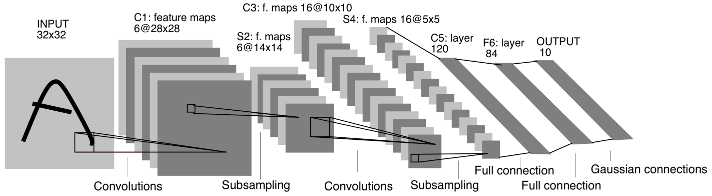


Fig. 2. Architecture of LeNet-5, a Convolutional Neural Network, here for digits recognition. Each plane is a feature map, i.e. a set of units whose weights are constrained to be identical. [2]

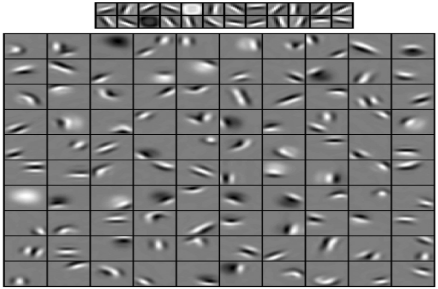
C1: conv. layer with 6 feature maps, 5 by 5 support, stride 1.
S2 (and S4): non-overlapping 2 by 2 blocks which equally sum values, mult by weight and add bias.
C3: conv. layer with 16 features, 5 by 5 support, partial connected.
C5: 120 features, 5 by 5 support, no stride; i.e. fully connected.
F6: fully connected, $W \in \mathbb{R}^{84\times 120}$.
CNNs trained on a data set typically work well on other, related, data sets with different labels by retraining just the last layer.
Why might this be? What do the conv. filters look like?

[2] `http://yann.lecun.com/exdb/publis/pdf/lecun-98.pdf`

Convolutional Deep Belief Networks(H. Lee et al. 11'[4])

We omit the details of this somewhat different architecture, which is stylistically similar to a deep CNN.

Figure 3. The first layer bases (top) and the second layer bases (bottom) learned from natural images. Each second layer basis (filter) was visualized as a weighted linear combination of the first layer bases.



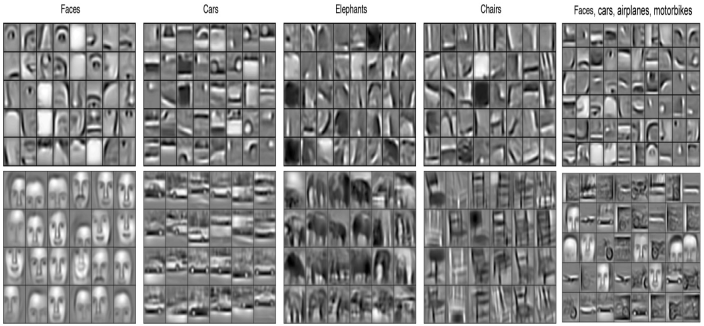
Display of the convolutional masks in layers 1 and 2, trained from Kyoto natural image database.[3] Note wavelet structures.

[3]http://eizaburo-doi.github.io/kyoto_natim/

[4]http://www.cs.utoronto.ca/~rgrosse/cacm2011-cdbn.pdf

Convolutional Deep Belief Networks(H. Lee et al. 11'[6])

Figure 4. Columns 1–4: the second layer bases (top) and the third layer bases (bottom) learned from specific object categories. Column 5: the second layer bases (top) and the third layer bases (bottom) learned from a mixture of four object categories (faces, cars, airplanes, motorbikes).

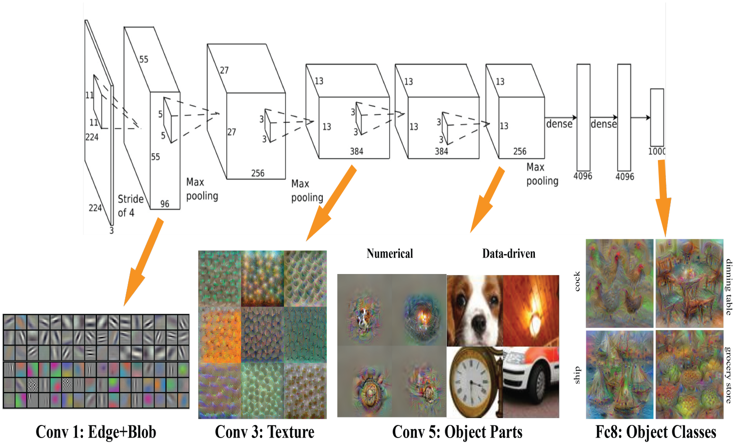


The third and fourth layers develop bases which represent features or objects, trained on CalTech 101 dataset.[5].

[5] http://www.vision.caltech.edu/Image_Datasets/Caltech101/

[6] http://www.cs.utoronto.ca/~rgrosse/cacm2011-cdbn.pdf

Deep CNN, AlexNet(Krizhevsky et al. 12'[7])



Images are those that maximize specific activation responses.

Layer 1 are masks, subsequent layers are their linear combinations.

[7] http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf

Deep CNN, VGG(Mahendran et al. 16'[8])

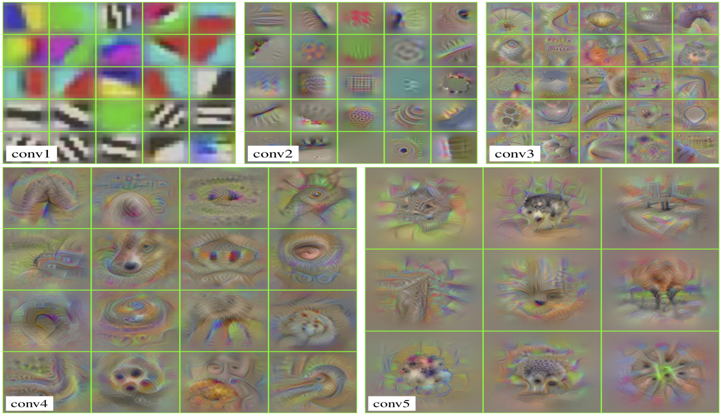


Figure 16: Activation maximization of the first filters of each convolutional layer in VGG-M.

Note, again we observe the same pattern, the initial filters are similar to Gabor/Wavelet filters and later layers are image components.

[8] `https://arxiv.org/abs/1512.02017`

Summary: similarity and importance of initial layers

We observe the initial layer of CNNs to be similar to one another, and to exhibit wavelet like representations. This is to be expected.

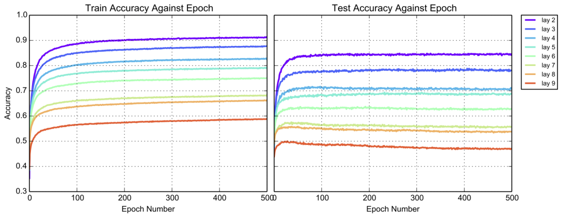


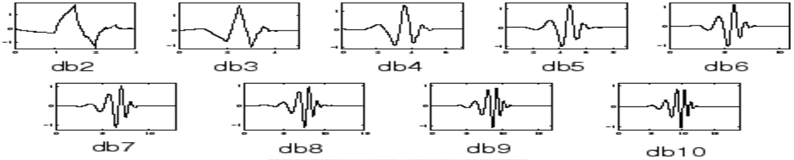
Figure 6: Demonstration of expressive power of remaining depth on MNIST. Here we plot train and test accuracy achieved by training exactly one layer of a fully connected neural net on MNIST. The different lines are generated by varying the hidden layer chosen to train. All other layers are kept frozen after random initialization.

Accuracy of a random network is improved most by training earlier layers (Raghu 16'[9]).

[9] `https://arxiv.org/pdf/1611.08083.pdf`

Wavelet, curvelet, and contourlet: fixed representations

Applied and computational harmonic analysis community developed representations with optimal approximation properties for piecewise smooth functions.



Most notable are the Daubechies wavelets and Curvelets/Contourlets pioneered by Candes and Donoho. While optimal, in a certain sense, for a specific class of functions, they can typically be improved upon for any particular data set.

Optimality of curvelets in 2D

Theorem (Candes and Donoho 02'[a])

[a] http://www.curvelet.org/papers/CurveEdges.pdf

Let f be a two dimensional function that is piecewise C^2 with a boundary that is also C^2. Let f_n^F, f_n^W, and f_n^C be the best approximation of f using n terms of the Fourier, Wavelet and Curvelet representation respectively. Then their approximation error satisfy $\|f - f_n^F\|_{L^2}^2 = \mathcal{O}(n^{-1/2})$, $\|f - f_n^W\|_{L^2}^2 = \mathcal{O}(n^{-1})$, and $\|f - f_n^C\|_{L^2}^2 = \mathcal{O}(n^{-2}\log^3(n))$; moreover, no fixed representation can have a rate exceeding $\mathcal{O}(n^{-2})$.

Near optimality of such representation suggest a good first layer.

Should be we deterministic or use learning?

The first layer of a net is seemingly the most important, and if we have good prior knowledge of the data we can probably guess near optimal candidate weights.

One can perform classification based on two layer net with:

layer 1: $h_2(x) = \sigma(W^{(1)}x + b^{(1)})$ where $W^{(1)}$ is a fixed transform of x to, say, the wavelet domain and $\sigma(\cdot)$ project to keep just the largest entries with hard or soft thresholding;

$$\sigma_{hard}(x;\tau) = \begin{cases} x & x > \tau \\ 0 & |x| \leq \tau \\ -x & x < -\tau \end{cases}, \quad \sigma_{soft}(x;\tau) = \begin{cases} x - \tau & x > \tau \\ 0 & |x| \leq \tau \\ -x + \tau & x < -\tau \end{cases}$$

layer2: $h_3 = \sigma(W^{(2)}h_2 + b^{(2)})$ with $W^{(2)}$ learned as the classifier based on the sparse codes h_2.

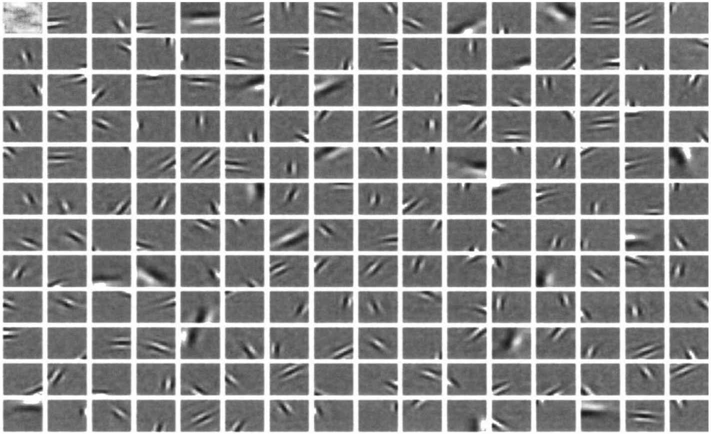
However, h_2 does not build in invariance we would desire in classification, such as dilation, rotation, translation, etc... Depth remains important to generate these.

Dictionary learning

While there are representations that are near optimal for realistic classes of functions, one can usually improve upon them for a particular data set; that is, one can learn a better dictionary for that data.
Let $Y \in \mathbb{R}^{m \times p}$ be a collection of p data elements in $\mathbb{R}^m$. Each data element y_i can be well represented by a dictionary $D \in \mathbb{R}^{m \times n}$ if there exists an x_i with at most k nonzeros such that $\|y_i - Dx_i\| \leq \epsilon(k)$. This can be combined in matrix notation as $\min_X \|Y - DX\|$ subject to $\|x_i\|_0 \leq k$ for $i = 1, \ldots, p$.
Note that solving for the optimal x_i for each y_i is in general NP hard, but for well behaved D it is easy.
Dictionary learning does a step further and learns the optimal D

$$\min_{D,X} \|Y - DX\| \quad \text{subject to} \quad \|x_i\|_0 \leq k, \ \|d_i\| = 1$$

Dictionary learned from natural scenes (Olshausen and Field 96'[10]



Note the similarity to curvelets and the first layer of deep CNNs.

[10] `https://www.nature.com/articles/381607a0.pdf`

Dictionary learning through ADMM

Alternating direction method of multipliers (ADMM) holds all but one component of a problem fixed and solves the other, then iterates through the variables to be solved for.
For dictionary learning this is iteratively solving:

$$\min_{X:\|x_i\|_0 \leq k} \|Y - DX\| \quad \text{then} \quad \min_{D:\|d_i\|=1} \|Y - DX\|$$

There are <u>many</u> methods for solving each of these subproblems.
Solving for X is more challenging, and will be our focus for now.
While better solutions exist, if X is held fixed one can solve for $YX^T = DXX^T$ as $X \in \mathbb{R}^{n\times p}$ for $p > n$ allowing $D = YX^T(XX^T)^{-1}$ followed by normalising the columns.

Coherence

- With $n > m$ the columns of $D \in \mathbb{R}^{m\times n}$ can't be orthogonal, we measure their dependence by the coherence of the columns.

$$\mu_2(D) := \max_{i\neq j} |d_i^* d_j|$$

- The collection of columns which are minimally coherent are called Grassman Frames and satisfy:

$$\mu_2(A_{m,n}) \geq \left(\frac{n-m}{m(n-1)}\right)^{1/2} \sim m^{-1/2}$$

- We can use coherence to analyse a number of algorithms to try and solve the sparse coding problem

$$\min_x \|x\|_0 \quad \text{subject to} \quad \|y_i - Dx_i\| \leq \tau$$

which in its worst case is NP-hard to solve.

One step thresholding

Input: y, D and k (number of non-zeros in output vector).
Algorithm: Set Λ the index set of the $k \le m$ largest in $|D^* b|$
Output the n-vector x whose entries are

$$x_\Lambda = (D_\Lambda^* D_\Lambda)^{-1} D_\Lambda^* y \quad \text{and} \quad x(i) = 0 \text{ for } i \notin \Lambda.$$

Theorem

Let $y = Dx_0$, with the columns of D having unit ℓ^2 norm, and

$$\|x_0\|_0 < \frac{1}{2}\left(\nu_\infty(x_0) \cdot \mu_2(D)^{-1} + 1\right),$$

then the Thresholding decoder with $k = \|x_0\|_0$ will return x_0, with $\nu_p(x) := \min_{j \in supp(x)} |x(j)|/\|x\|_p$.

One step thresholding (proof)

Proof.

With $y = Dx_0$, denote $w = D^*b = D^*Dx_0$.
The i^{th} entry in w is equal to $w_i = \sum_{j \in \text{supp}(x_0)} x_0(j) d_i^* d_j$.
For $i \notin \text{supp}(x_0)$ the magnitude of w_i is bounded above as:

$$|w_i| \leq \sum_{j \in \text{supp}(x_0)} |x_0(j)| \cdot |d_i^* d_j| \leq k\mu_2(D)\|x_0\|_\infty .$$

For $i \in \text{supp}(x_0)$ the magnitude of w_i is bounded below as:

$$\begin{aligned} |w_i| &\geq |x_0(i)| - \left| \sum_{j \in \text{supp}(x_0), j \neq i} x_0(j) d_i^* d_j \right| \\ &\geq |x_0(i)| - (k-1)\mu_2(D)\|x_0\|_\infty . \end{aligned}$$

Recovery if $\max_{i \notin \text{supp}(x_0)} |w_i| < \min_{i \in \text{supp}(x_0)} |w_i|$. □

Matching Pursuit (Tropp 04'[11])

Input: y, D and k (number of nonzeros in output vector).
Algorithm: Let $r^j := y - Dx^j$.
Set $x^0 = 0$, and let $i := \mathrm{argmax}_\ell |a_\ell^* r^j|$ and define
$x^{j+1} = x^j + (d_i^* r^j)e_i$ where e_i is the i^{th} coordinate vector.
Output x^j when a termination criteria is obtained.

Theorem

Let $y = Dx_0$, with the columns of D having unit ℓ^2 norm, and

$$\|x_0\|_{\ell^0} < \frac{1}{2}\left(\mu_2(D)^{-1} + 1\right),$$

then Matching Pursuit will have $supp(x^j) \subseteq supp(x_0)$ for all j.

$*$ Preferable over one step thresholding: no dependence on $\nu_p(x_0)$.

[11] https://ieeexplore.ieee.org/document/1337101

Matching Pursuit (proof)

Proof.

Assume $\text{supp}(x^j) \subset \text{supp}(x_0)$ for some j, which is true for $j = 0$.
Let $r^j = y - A_{m,n}x^j$, and $w_i = \sum_{\ell \in \text{supp}(x_0)} (x_0 - x^j)(\ell) \cdot d_i^* d_\ell$.
For $i \notin \text{supp}(x_0)$ the magnitude of w_i is bounded above as:

$$|w_i| \leq \sum_{\ell \in \text{supp}(x_0)} |(x_0 - x^j)(\ell)| \cdot |d_i^* d_\ell| \leq k\mu_2(D) \| |x_0 - x^j\|_\infty .$$

For $i \in \text{supp}(x_0)$ the magnitude of w_i is bounded below as:

$$\begin{aligned} |w_i| &\geq |(x_0 - x^j)(i)| - \left| \sum_{\ell \in \text{supp}(x_0), \ell \neq i} (x_0 - x^j)(\ell) \cdot d_i^* d_\ell \right| \\ &\geq |(x_0 - x^j)(i)| - (k-1)\mu_2(D) \|x_0 - x^j\|_\infty . \end{aligned}$$

Recovery if $\max_{i \in \text{supp}(x_0)} |w_i| > \max_{i \notin \text{supp}(x_0)} |w_i|$.

□

Orthogonal Matching Pursuit (Tropp 04'[12])

Input: y, D and k (number of nonzeros in output vector).
Algorithm: Let $r^j := y - Dx^j$.
Set $x^0 = 0$ and Λ^0 to be the empty set, and set $j = 0$.
Let $r^j := y - Dx^j$, $i := \text{argmax}_\ell |d_\ell^* r^j|$, and $\Lambda^{j+1} = i \bigcup \Lambda^j$.
Set $x_{\Lambda^{j+1}}^{j+1} = (D_{\Lambda^{j+1}}^* D_{\Lambda^{j+1}})^{-1} D_{\Lambda^{j+1}}^* y$
and $x^{j+1}(\ell) = 0$ for $\ell \notin \Lambda^{j+1}$, and set $j = j + 1$.
Output x^j when a termination criteria is obtained.

Theorem

Let $y = Dx_0$, with the columns of D having unit ℓ^2 norm, and

$$\|x_0\|_{\ell^0} < \frac{1}{2}\left(\mu_2(D)^{-1} + 1\right),$$

then after $\|x_0\|_{\ell^0}$ steps, Orthogonal Matching Pursuit recovers x_0.

$*$ Proof, same as Matching Pursuit. Finite number of steps.

[12] https://ieeexplore.ieee.org/document/1337101

ℓ^1-regularization (Tropp 06'[13])

Input: y and D.
"Algorithm": Return argmin$\|x\|_1$ subject to $y = Dx$.

Theorem
Let $y = A_{m,n}x_0$, with

$$\|x_0\|_{\ell^0} < \frac{1}{2}\left(\mu_2(D)^{-1} + 1\right),$$

then the solution of ℓ^1-regularization is x_0.

$\ast$ Preferable over OMP: faster if use good ℓ^1 solver.

[13] `http://users.cms.caltech.edu/~jtropp/papers/Tro06-Just-Relax.pdf`

ℓ^1-regularization (proof, page 1)

Proof.

Let $\Lambda_0 := supp(x_0)$ and $\Lambda_1 := supp(x_1)$ with $y = Dx_0 = Dx_1$, and $\exists i$ with $i \in \Lambda_1$ with $i \notin \Lambda_0$.

Note that because $y = D_{\Lambda_0} x_0 = D_{\Lambda_1} x_1$,

$$\begin{aligned}\|x_0\|_1 &= \|(D^*_{\Lambda_0} D_{\Lambda_0})^{-1} D^*_{\Lambda_0} D_{\Lambda_0} x_0\|_1 \\ &= \|(D^*_{\Lambda_0} D_{\Lambda_0})^{-1} D^*_{\Lambda_0} y\|_1 \\ &= \|(D^*_{\Lambda_0} D_{\Lambda_0})^{-1} D^*_{\Lambda_0} D_{\Lambda_1} x_1\|_1.\end{aligned}$$

Establish bounds on $(D^*_{\Lambda_0} D_{\Lambda_0})^{-1} D^*_{\Lambda_0} d_i$.

To establish proof need bounds for $i \in \Lambda$ and $i \notin \Lambda$.

For $i \in \Lambda_0$: $\|(D^*_{\Lambda_0} D_{\Lambda_0})^{-1} D^*_{\Lambda_0} d_i\|_1$
$= \|(D^*_{\Lambda_0} D_{\Lambda_0})^{-1} D^*_{\Lambda_0} D_{\Lambda_0} e_i\|_1 = \|e_i\|_1 = 1$

□

ℓ^1-regularization (proof, page 2)

Proof.

For any $i \notin \Lambda_0$ we establish the bound in two parts; first,

$$\|D_{\Lambda_0}^* d_i\|_1 \leq \sum_{\ell \in \Lambda_0} |d_\ell^* d_i| \leq k\mu_2(D).$$

Noting $D_{\Lambda_0}^* D_{\Lambda_0} = I_{k,k} + B$ where $B_{i,i} = 0$ and $|B_{i,j}| \leq \mu_2(D)$, then

$$\|(I_{k,k}+B)^{-1}\|_1 = \left\|\sum_{\ell=0}^{\infty}(-B)^\ell\right\|_1 \leq \sum_{\ell=0}^{\infty} \|B\|_1^\ell = \frac{1}{1-\|B\|_1} \leq \frac{1}{1-(k-1)\mu}$$

Therefore, for $i \notin \Lambda_0$:

$$\|(D_{\Lambda_0}^* D_{\Lambda_0})^{-1} D_{\Lambda_0}^* d_i\|_1 \leq \frac{k\mu_2(D)}{(1-(k-1)\mu_2(D))} < 1$$

□

ℓ^1-regularization (proof, page 3)

Proof.

Proof concludes through triangle inequality and use that:

- For $i \in \Lambda_0$: $\|(D_{\Lambda_0}^* D_{\Lambda_0})^{-1} D_{\Lambda_0}^* d_i\|_1 = 1$
- For $i \notin \Lambda_0$: $\|(D_{\Lambda_0}^* D_{\Lambda_0})^{-1} D_{\Lambda_0}^* d_i\|_1 < 1$
- And $\exists i$ with $i \in \Lambda_1$ and $i \notin \Lambda_0$.

Then,

$$\begin{aligned} \|x_0\|_1 &= \left\| \sum_{i \in \Lambda_1} (D_{\Lambda_0}^* D_{\Lambda_0})^{-1} D_{\Lambda_0}^* d_i x_1(i) \right\|_1 \\ &\leq \sum_{i \in \Lambda_1} |x_1(i)| \cdot \left\| (D_{\Lambda_0}^* D_{\Lambda_0})^{-1} D_{\Lambda_0}^* d_i \right\|_1 \\ &< \sum_{i \in \Lambda_1} |x_1(i)| = ||x_1\|_1. \end{aligned}$$

□

But, is the solution even unique?

The sparsity of the sparsest vector in the nullspace of D,

$$spark(D) := \min_z \|z\|_{\ell^0} \quad \text{subject to} \quad Dz = 0.$$

Theorem (Spark and Coherence)

$$spark(D) \geq \min(m+1, \mu_2(D)^{-1} + 1)$$

If $\|x_0\| < (\mu_2(D)^{-1} + 1)/2$ *unique satisfying* $y = Dx_0$.

Proof.

Gershgorin disc theorem for $D_\Lambda^* D_\Lambda$ with $|\Lambda| = k$:
1 on diagonal, off diagonals bounded by $\mu_2(D)$.
If $k < \mu_2(D)^{-1} + 1$, smallest singular value of $D_\Lambda^* D_\Lambda$ is > 0 □