K-SVD FOR DUMMIES

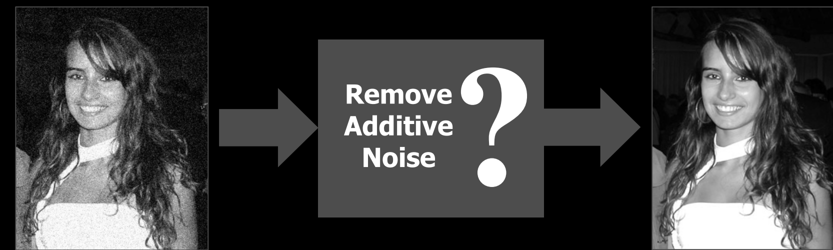
An Introduction to Sparse Representation and the K-SVD Algorithm

Ron Rubinstein
The CS Department
The Technion – Israel Institute of technology
Haifa 32000, Israel

Friedrich-Alexander-Universität Erlangen-Nürnberg
University of Erlangen - Nürnberg
April 2008

Noise Removal ?

Our story begins with image denoising …



- Practical application
- A convenient platform (being the simplest inverse problem) for testing basic ideas in image processing.

Denoising By Energy Minimization

Many of the proposed denoising algorithms are related to the minimization of an energy function of the form

$$f(\underline{x}) = \frac{1}{2}\left\|\underline{x}-\underline{y}\right\|_2^2 + \Pr(\underline{x})$$

Sanity (relation to measurements)

Prior or regularization

$\underline{y}$: Given measurements

$\underline{x}$: Unknown to be recovered

- This is in-fact a Bayesian point of view, adopting the Maximum-Aposteriori Probability (MAP) estimation.
- Clearly, the wisdom in such an approach is within the choice of the prior – **modeling the images** of interest.

Thomas Bayes
1702 - 1761

The Evolution Of Pr($\underline{x}$)

During the past several decades we have made all sort of guesses about the prior Pr($\underline{x}$) for images:

$\Pr(\underline{x}) = \lambda\left\|\underline{x}\right\|_2^2$ **Energy**

$\Pr(\underline{x}) = \lambda\left\|\mathbf{L}\underline{x}\right\|_2^2$ **Smoothness**

$\Pr(\underline{x}) = \lambda\left\|\mathbf{L}\underline{x}\right\|_{\mathbf{W}}^2$ **Adapt+ Smooth**

$\Pr(\underline{x}) = \lambda\rho\left\{\mathbf{L}\underline{x}\right\}$ **Robust Statistics**

$\Pr(\underline{x}) = \lambda\left\|\nabla\underline{x}\right\|_1$ **Total-Variation**

$\Pr(\underline{x}) = \lambda\left\|\mathbf{W}\underline{x}\right\|_1$ **Wavelet Sparsity**

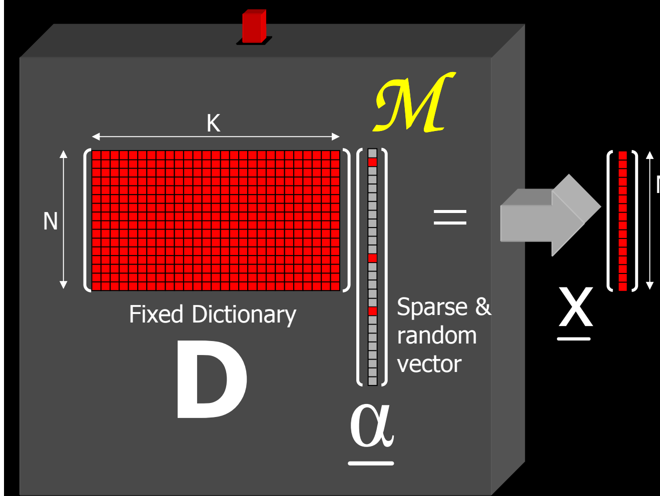
$\Pr(\underline{x}) = \lambda\left\|\mathbf{B}\underline{x}\right\|_1$ **Bilateral Filter**

$\Pr(\underline{x}) = \lambda\left\|\underline{\alpha}\right\|_0^0$ for $\underline{x} = \mathbf{D}\underline{\alpha}$ **Sparse & Redundant**

Agenda

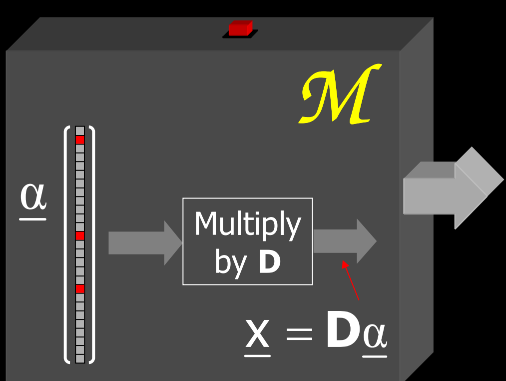
1. **A Visit to *Sparseland***
   Introducing sparsity & overcompleteness
2. Transforms & Regularizations
   How & why should this work?
3. What about the dictionary?
   The quest for the origin of signals
4. Putting it all together
   Image filling, denoising, compression, …

Generating Signals in *Sparseland*



- Every column in **D** (dictionary) is a prototype signal (atom).
- The vector $\underline{\alpha}$ is generated randomly with few non-zeros in random locations and with random values.

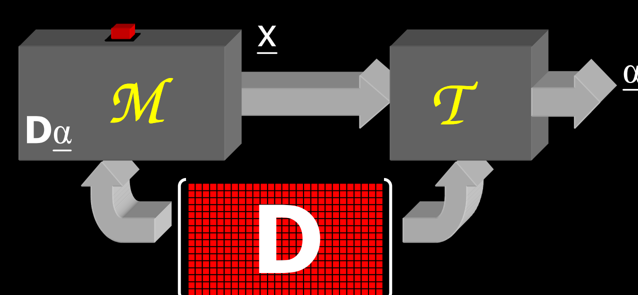
Sparseland Signals Are Special



- **Simple:** Every signal is built as a linear combination of a **few** atoms from the dictionary **D**.
- **Effective:** Recent works adopt this model and successfully deploy it to applications.
- **Empirically established:** Neurological studies show similarity between this model and early vision processes. [Olshausen & Field ('96)]

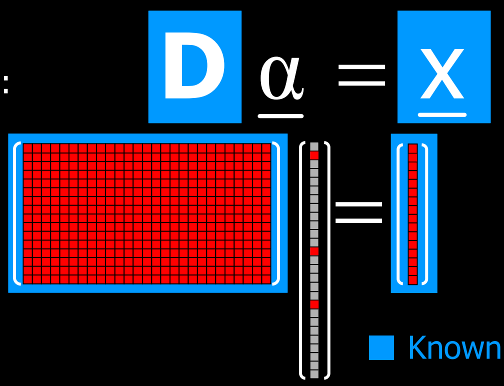
Transforms in *Sparseland* ?

- Assume that $\underline{x}$ is known to emerge from $\mathcal{M}$.
- How about "Given $\underline{x}$, find the $\underline{\alpha}$ that generated it in $\mathcal{M}$ " ?



Problem Statement

We need to solve an under-determined linear system of equations:

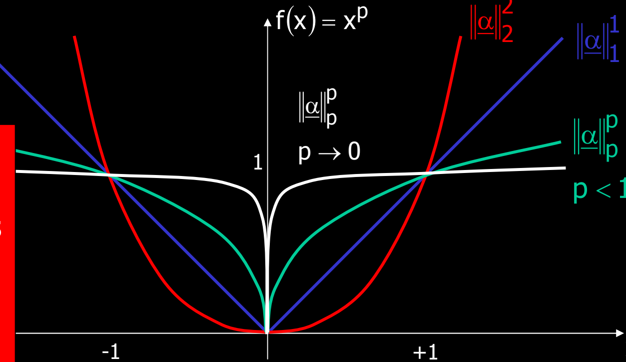


- Among all (infinitely many) possible solutions we want the sparsest !!
- We will measure sparsity using the L_0 "norm": $\rightarrow \|\underline{\alpha}\|_0$

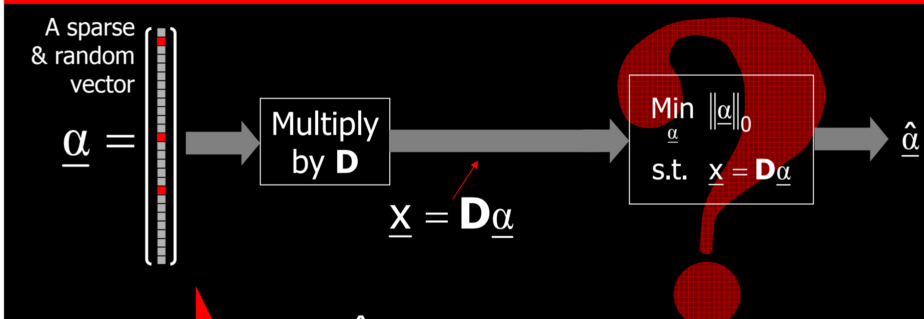
Measure of Sparsity?

$$\|\underline{x}\|_p^p = \sum_{j=1}^{k} |x_j|^p$$

As $p \rightarrow 0$ we get a count of the non-zeros in the vector $\rightarrow \|\underline{\alpha}\|_0$



Where We Are

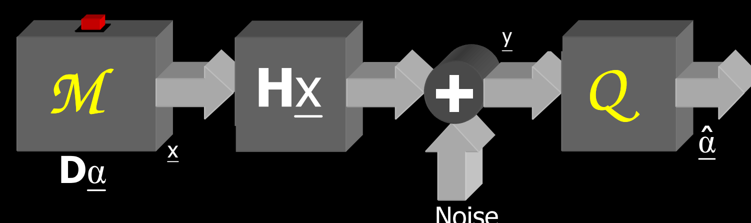


3 Major Questions

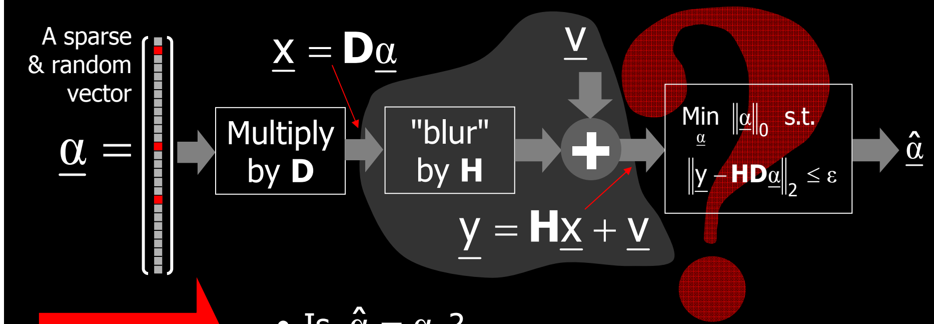
- Is $\hat{\underline{\alpha}} = \underline{\alpha}$?
- NP-hard: practical ways to get $\hat{\underline{\alpha}}$?
- How do we know **D**?

Inverse Problems in *Sparseland* ?

- Assume that $\underline{x}$ is known to emerge from $\mathcal{M}$.
- Suppose we observe $\underline{y} = \mathbf{H}\underline{x} + \underline{v}$, a degraded and noisy version of $\underline{x}$ with $\|\underline{v}\|_2 \le \varepsilon$. How do we recover $\underline{x}$?
- How about "find the $\underline{\alpha}$ that generated $\underline{y}$" ?



Inverse Problem Statement

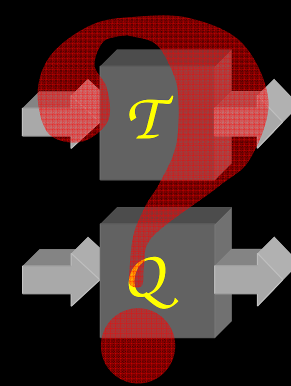


3 Major Questions (again!)

- Is $\hat{\underline{\alpha}} = \underline{\alpha}$?
- How can we compute $\hat{\underline{\alpha}}$?
- What **D** should we use?

Agenda

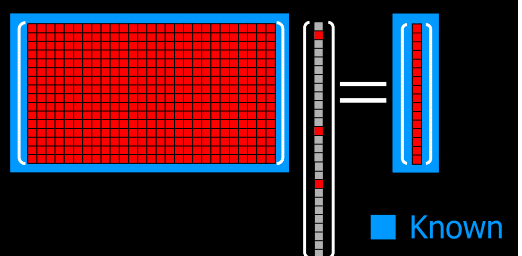
1. A Visit to *Sparseland*
   Introducing sparsity & overcompleteness
2. **Transforms & Regularizations**
   How & why should this work?
3. What about the dictionary?
   The quest for the origin of signals
4. Putting it all together
   Image filling, denoising, compression, …



The Sparse Coding Problem

Our dream for now: Find the sparsest solution to

$$\mathbf{D}\underline{\alpha} = \underline{x}$$

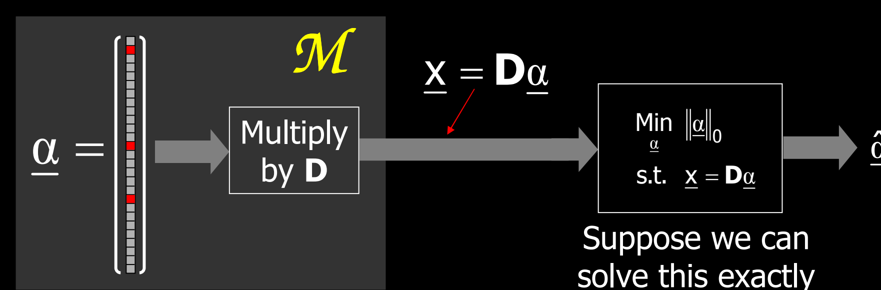


Put formally,

$$\underset{\underline{\alpha}}{\text{Min}} \|\underline{\alpha}\|_0 \quad \text{s.t.} \quad \underline{x} = \mathbf{D}\underline{\alpha}$$

known

Question 1 – Uniqueness?



Suppose we can solve this exactly

Why should we necessarily get $\hat{\underline{\alpha}} = \underline{\alpha}$?

It might happen that eventually $\|\hat{\underline{\alpha}}\|_0 < \|\underline{\alpha}\|_0$.

Matrix "Spark"

Definition: Given a matrix **D**, σ=Spark{**D**} is the smallest number of columns that are linearly dependent.

Donoho & Elad ('02)

- By definition, if $\mathbf{D}\underline{v}=0$ then $\|\underline{v}\|_0 \geq \sigma$
- Say I have $\underline{\alpha}_1$ and you have $\underline{\alpha}_2$, and the two are different representations of the same $\underline{x}$:

$$\underline{x} = \mathbf{D}\underline{\alpha}_1 = \mathbf{D}\underline{\alpha}_2 \quad \Rightarrow \quad \mathbf{D}(\underline{\alpha}_1 - \underline{\alpha}_2) = \underline{0}$$

$$\Rightarrow \|\underline{\alpha}_1 - \underline{\alpha}_2\|_0 \geq \sigma$$

Uniqueness Rule

- Now, what if my $\underline{\alpha}_1$ satisfies $\|\underline{\alpha}_1\|_0 < \frac{\sigma}{2}$?
- The rule $\|\underline{\alpha}_1 - \underline{\alpha}_2\|_0 \geq \sigma$ implies that $\|\underline{\alpha}_2\|_0 > \frac{\sigma}{2}$!

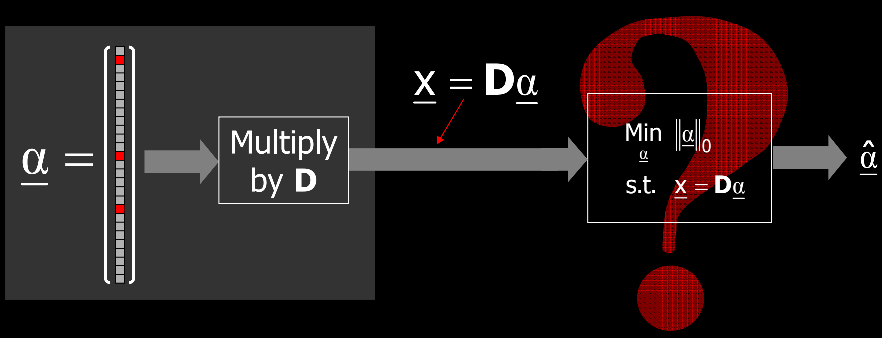
Uniqueness

Donoho & Elad ('02)

If we have a representation that satisfies $\frac{\sigma}{2} > \|\underline{\alpha}\|_0$ then necessarily it is the sparsest.

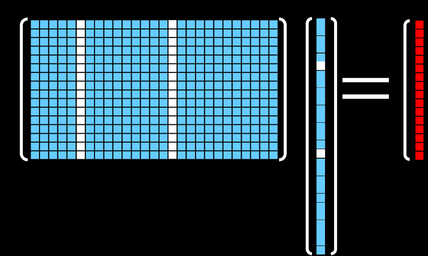
So, if $\mathcal{M}$ generates signals using "sparse enough" $\underline{\alpha}$, the solution of $P_0 : \min_{\underline{\alpha}} \|\underline{\alpha}\|_0$ s.t. $\underline{x} = \mathbf{D}\underline{\alpha}$ will find them exactly.

Question 2 – Practical P_0 Solver?



Are there reasonable ways to find $\hat{\underline{\alpha}}$?

Matching Pursuit (MP) Mallat & Zhang (1993)



- The MP is a greedy algorithm that finds one atom at a time.
- Step 1: find the one atom that best matches the signal.
- Next steps: given the previously found atoms, find the next **one** to best fit …
- The Orthogonal MP (OMP) is an improved version that re-evaluates the coefficients after each round.

Basis Pursuit (BP) Chen, Donoho, & Saunders (1995)

Instead of solving

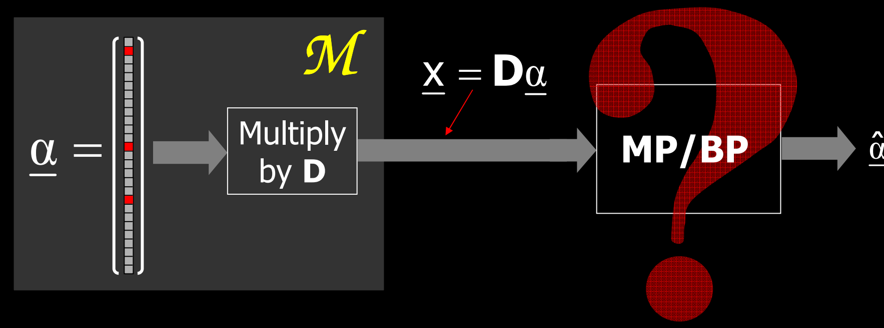
$$\min_{\underline{\alpha}} \|\underline{\alpha}\|_0 \ \text{ s.t. } \ \underline{x} = \mathbf{D}\underline{\alpha}$$

Solve this:

$$\min_{\underline{\alpha}} \|\underline{\alpha}\|_1 \ \text{ s.t. } \ \underline{x} = \mathbf{D}\underline{\alpha}$$

- The newly defined problem is convex (linear programming).
- Very efficient solvers can be deployed:
  - Interior point methods [Chen, Donoho, & Saunders (`95)] ,
  - Iterated shrinkage [Figuerido & Nowak (`03), Daubechies, Defrise, & Demole ('04), Elad (`05), Elad, Matalon, & Zibulevsky (`06)].

Question 3 – Approx. Quality?



How effective are MP/BP ?

BP and MP Performance

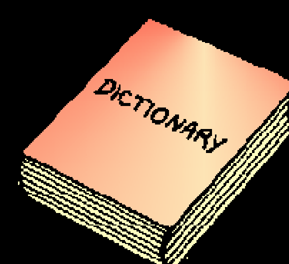
Donoho & Elad ('02)
Gribonval & Nielsen ('03)
Tropp ('03)
Temlyakov ('03)

Given a signal $\underline{x}$ with a representation $\underline{x} = \mathbf{D}\underline{\alpha}$, if $\|\underline{\alpha}\|_0 < (\textit{some threshold})$ then BP and MP are **guaranteed** to find it.

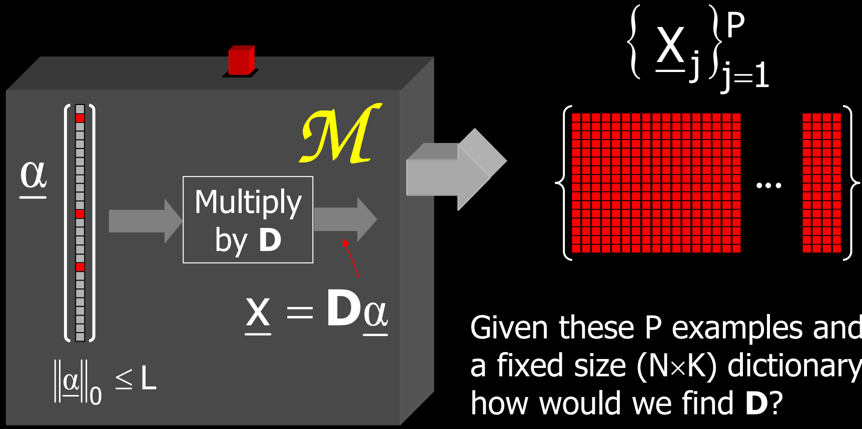
- MP and BP are different in general (hard to say which is better).
- The above results correspond to the worst-case.
- Average performance results available too, showing much better bounds [Donoho (`04), Candes et.al. (`04), Tanner et.al. (`05), Tropp et.al. (`06)].
- Similar results for general inverse problems [Donoho, Elad & Temlyakov (`04), Tropp (`04), Fuchs (`04), Gribonval et. al. (`05)].

Agenda

1. A Visit to *Sparseland*
   Introducing sparsity & overcompleteness
2. Transforms & Regularizations
   How & why should this work?
3. **What about the dictionary?**
   The quest for the origin of signals
4. Putting it all together
   Image filling, denoising, compression, …

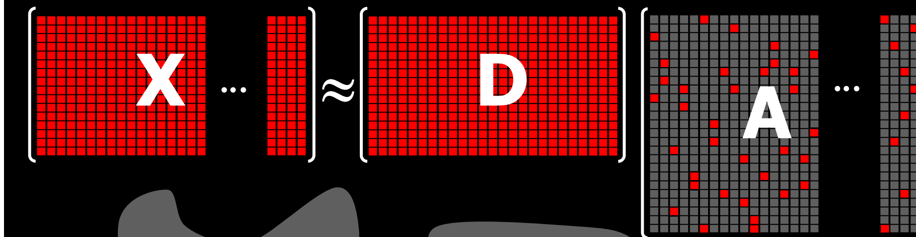


Problem Setting



Given these P examples and a fixed size (N×K) dictionary, how would we find **D**?

The Objective Function



$$\min_{\mathbf{D},\mathbf{A}} \|\mathbf{D}\mathbf{A}-\mathbf{X}\|_F^2 \quad s.t. \quad \forall j, \|\underline{\alpha}_j\|_0 \le L$$

The examples are linear combinations of atoms from **D**

Each example has a sparse representation with no more than L atoms

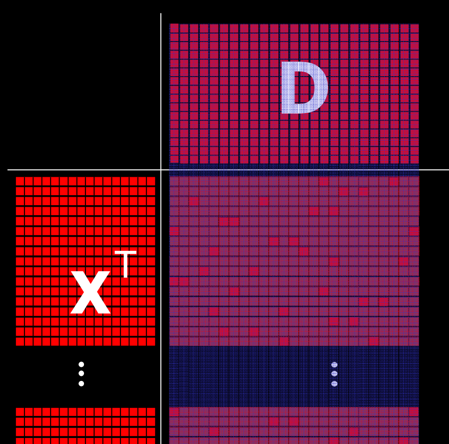
(N,K,L are assumed known, **D** has normalized columns)

K–SVD – An Overview

Initialize **D**

Sparse Coding
Use MP or BP

Dictionary Update
Column-by-Column by **SVD computation**



Aharon, Elad & Bruckstein ('04)

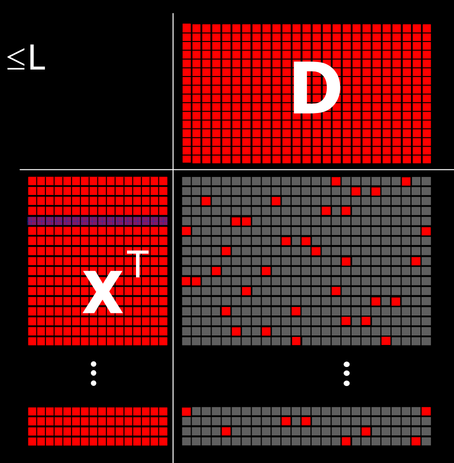
K–SVD: Sparse Coding Stage

$$\min_{\mathbf{A}} \|\mathbf{D}\mathbf{A}-\mathbf{X}\|_F^2 \quad \text{s.t.} \quad \forall j, \|\underline{\alpha}_j\|_0 \le L$$

For the j[th] example we solve

$$\min_{\underline{\alpha}} \|\mathbf{D}\underline{\alpha}-\underline{x}_j\|_2^2 \quad \text{s.t.} \quad \|\underline{\alpha}\|_0 \le L$$

Ordinary Sparse Coding !



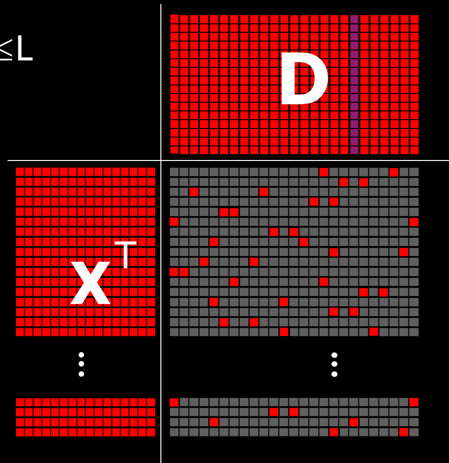
K–SVD: Dictionary Update Stage

$$\min_{\mathbf{D}} \left\|\mathbf{D}\mathbf{A}-\mathbf{X}\right\|_F^2 \quad \text{s.t.} \quad \forall j, \left\|\underline{\alpha}_j\right\|_0 \le L$$

For the kth atom we solve

$$\min_{\underline{d}_k} \left\|\underline{d}_k \underline{a}_k^T - \mathbf{E}_k\right\|_F^2$$

$\mathbf{E}_k = \sum_{j \neq k} \underline{d}_j \underline{a}_j^T - \mathbf{X}$ **(the residual)**



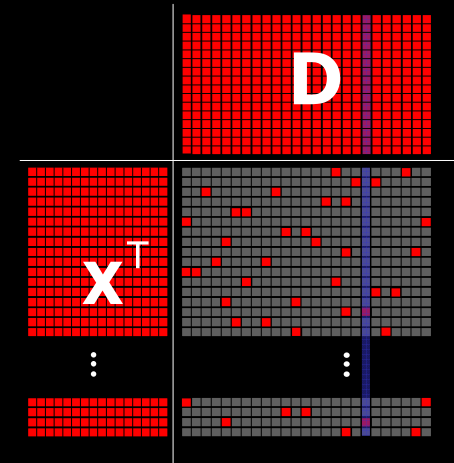
K–SVD Dictionary Update Stage

$$\min_{\underline{d}_k} \left\|\underline{d}_k \underline{a}_k^T - \mathbf{E}_k\right\|_F^2$$

We can do better than this

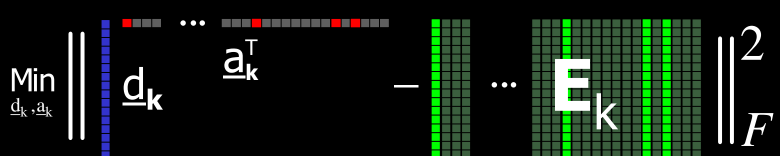
$$\min_{\underline{d}_k, \underline{a}_k} \left\|\underline{d}_k \underline{a}_k^T - \mathbf{E}_k\right\|_F^2$$

But wait! What about sparsity?



K–SVD Dictionary Update Stage

We want to solve:



Only some of the examples use column $\underline{d}_k$!

When updating $\underline{a}_k$, only recompute the coefficients corresponding to those examples

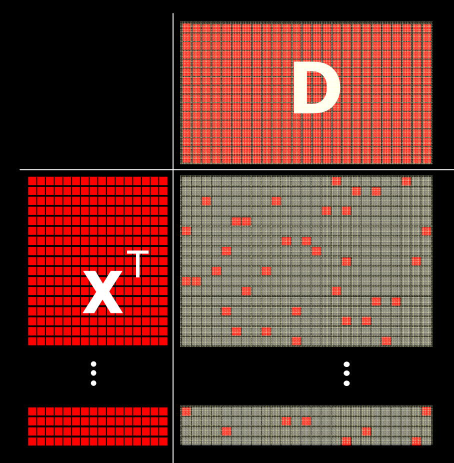
Solve with **SVD!**

The K–SVD Algorithm – Summary

Initialize **D**

Sparse Coding
Use MP or BP

Dictionary Update
Column-by-Column by **SVD computation**



Agenda

1. A Visit to *Sparseland*
   Introducing sparsity & overcompleteness
2. Transforms & Regularizations
   How & why should this work?
3. What about the dictionary?
   The quest for the origin of signals
4. **Putting it all together**
   Image filling, denoising, compression, …

Image Inpainting: Theory

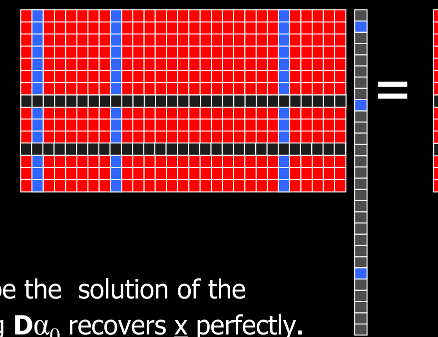
- Assumption: the signal $\underline{x}$ was created by $\underline{x}=\mathbf{D}\underline{\alpha}_0$ with a very sparse $\underline{\alpha}_0$.
- Missing values in $\underline{x}$ imply missing rows in this linear system.
- By removing these rows, we get

$$\tilde{\mathbf{D}}\underline{\alpha}_0 = \underline{\tilde{x}}$$

- Now solve

$$\underset{\underline{\alpha}}{\text{Min}} \|\underline{\alpha}\|_0 \quad \text{s.t.} \quad \underline{\tilde{x}} = \tilde{\mathbf{D}}\underline{\alpha}$$

$$\mathbf{D}\underline{\alpha}_0 = \underline{x}$$



- If $\underline{\alpha}_0$ was sparse enough, it will be the solution of the above problem! Thus, computing $\mathbf{D}\underline{\alpha}_0$ recovers $\underline{x}$ perfectly.

Inpainting: The Practice

- We define a diagonal mask operator **W** representing the lost samples, so that

$$\underline{y} = \mathbf{W}\underline{x} + \underline{v} \qquad w_{i,i} \in \{0,1\}$$

- Given $\underline{y}$, we try to recover the representation of $\underline{x}$, by solving

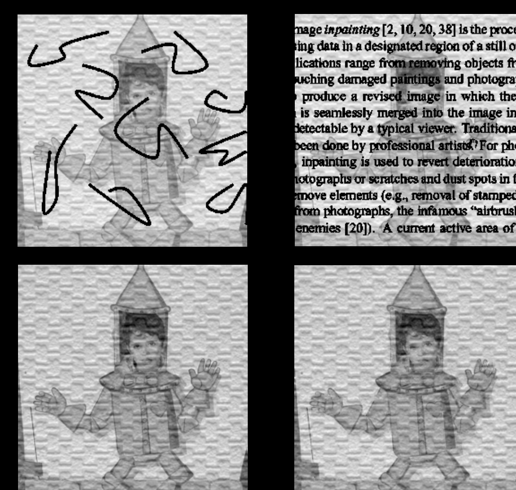
$$\hat{\underline{\alpha}} = \underset{\underline{\alpha}}{\text{ArgMin}} \|\underline{\alpha}\|_0 \quad \text{s.t.} \quad \|\underline{y} - \mathbf{W}\mathbf{D}\underline{\alpha}\|_2 \le \varepsilon \quad \Rightarrow \quad \hat{\underline{x}} = \mathbf{D}\hat{\underline{\alpha}}$$

- We use a dictionary that is the sum of two dictionaries, to get an effective representation of both texture and cartoon contents. This also leads to image separation [Elad, Starck, & Donoho ('05)]

Inpainting Results

Source

Outcome



Dictionary: Curvelet (cartoon) + Global DCT (texture)

Inpainting Results

Source

Outcome

Dictionary: Curvelet (cartoon) + Overlapped DCT (texture)

Inpainting Results

Denoising: Theory and Practice

- Given a noisy image $\underline{y}$, we can clean it by solving

$$\hat{\underline{\alpha}} = \underset{\underline{\alpha}}{\text{ArgMin}} \|\underline{\alpha}\|_0 \quad \text{s.t.} \quad \|\underline{y} - \mathbf{D}\underline{\alpha}\|_2 \le \varepsilon \quad \Rightarrow \quad \hat{\underline{x}} = \mathbf{D}\hat{\underline{\alpha}}$$

- Can we use the K-SVD dictionary?
- With K-SVD, we cannot train a dictionary for an entire image. How do we go from local treatment of patches to a global prior?
- Solution: force shift-invariant sparsity – for each NxN patch of the image, including overlaps.

From Local to Global Treatment

$$\hat{\underline{\alpha}} = \underset{\underline{\alpha}}{\text{ArgMin}} \|\underline{\alpha}\|_0 \quad \text{s.t.} \quad \|\underline{y} - \mathbf{D}\underline{\alpha}\|_2 \le \varepsilon \quad \Rightarrow \quad \hat{\underline{x}} = \mathbf{D}\hat{\underline{\alpha}}$$

For patches, our MAP penalty becomes

$$\hat{\underline{x}} = \underset{\underline{x},\{\underline{\alpha}_{ij}\}_{ij}}{\text{ArgMin}} \frac{1}{2}\|\underline{x} - \underline{y}\|_2^2 + \mu \sum_{ij} \|\mathbf{R}_{ij}\underline{x} - \mathbf{D}\underline{\alpha}_{ij}\|_2^2 \quad \text{s.t.} \quad \|\underline{\alpha}_{ij}\|_0 \le L$$

Extracts the (i,j)th patch

Our prior

What Data to Train On?

Option 1:

- Use a database of images: works quite well (~0.5-1dB below the state-of-the-art)

Option 2:

- Use the corrupted image itself !
- Simply sweep through all NxN patches (with overlaps) and use them to train
- Image of size 1000x1000 pixels ➡ ~10^6 examples to use – more than enough.
- This works much better!

Image Denoising: The Algorithm

$$\hat{\underline{x}} = \underset{\underline{x},\{\underline{\alpha}_{ij}\}_{ij},\mathbf{D}}{\text{ArgMin}} \frac{1}{2}\|\underline{x}-\underline{y}\|_2^2 + \mu\sum_{ij}\|\mathbf{R}_{ij}\underline{x}-\mathbf{D}\underline{\alpha}_{ij}\|_2^2 \quad \text{s.t.} \quad \|\underline{\alpha}_{ij}\|_0 \le L$$

$\underline{x}$ and **D** known | $\underline{x}$ and α_{ij} known | **D** and α_{ij} known

Compute α_{ij} per patch

$$\underline{\alpha}_{ij} = \underset{\underline{\alpha}}{\text{Min}} \|\mathbf{R}_{ij}\underline{x}-\mathbf{D}\underline{\alpha}\|_2^2 \quad \text{s.t.} \quad \|\underline{\alpha}_{ij}\|_0 \le L$$

using matching pursuit

Compute **D** to minimize

$$\underset{\mathbf{D}}{\text{Min}} \sum_{ij} \|\mathbf{R}_{ij}\underline{x}-\mathbf{D}\underline{\alpha}\|_2^2$$

using SVD, updating one column at a time

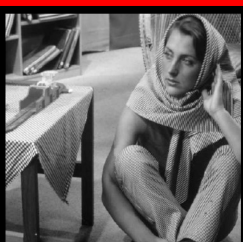
K-SVD

Compute $\underline{x}$ by

$$\underline{x} = \left[\mathrm{I} + \mu\sum_{ij}\mathbf{R}_{ij}^T\mathbf{R}_{ij}\right]^{-1}\left[\underline{y} + \mu\sum_{ij}\mathbf{R}_{ij}^T\mathbf{D}\underline{\alpha}_{ij}\right]$$

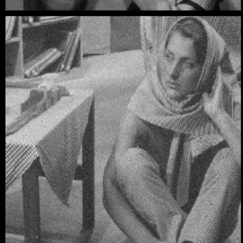
which is a simple averaging of shifted patches

Denoising Results



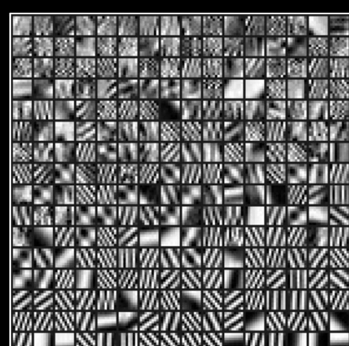
Source

Result 30.829dB



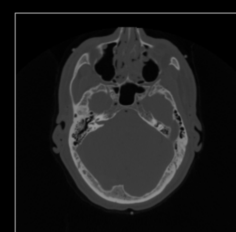
Noisy image

$\text{PSNR} = 22.1\text{dB}$

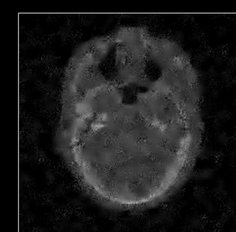


Obtained dictionary after 10 iterations

Denoising Results: 3D

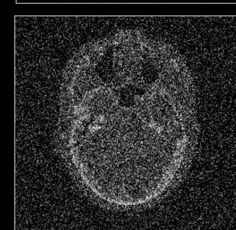


Source:
Vis. Male Head
(Slice #137)

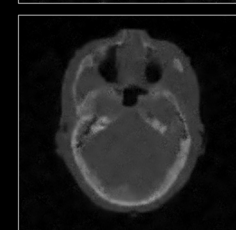


2d-KSVD:

PSNR=27.3dB



PSNR=12dB

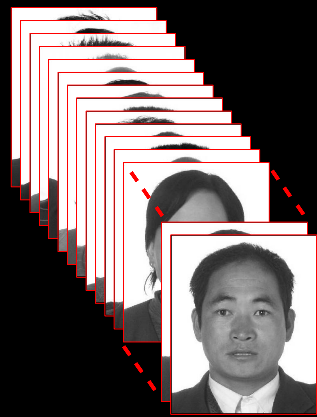


3d-KSVD:

PSNR=32.4dB

Image Compression

- Problem: compressing photo-ID images.
- General purpose methods (JPEG, JPEG2000) do not take into account the specific family.
- By adapting to the image-content, better results can be obtained.



Compression: The Algorithm

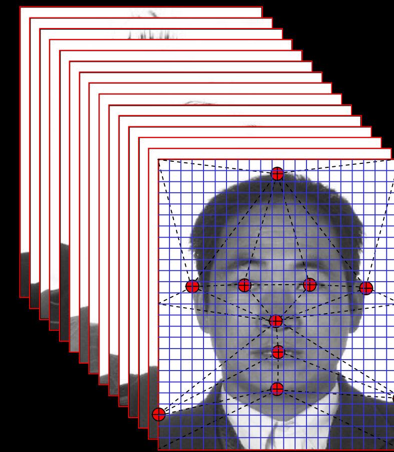
Training:

1. Detect main features and align the images to a common reference (20 parameters)
2. Divide each image to disjoint 15x15 patches, and for each compute a unique dictionary

Compression:

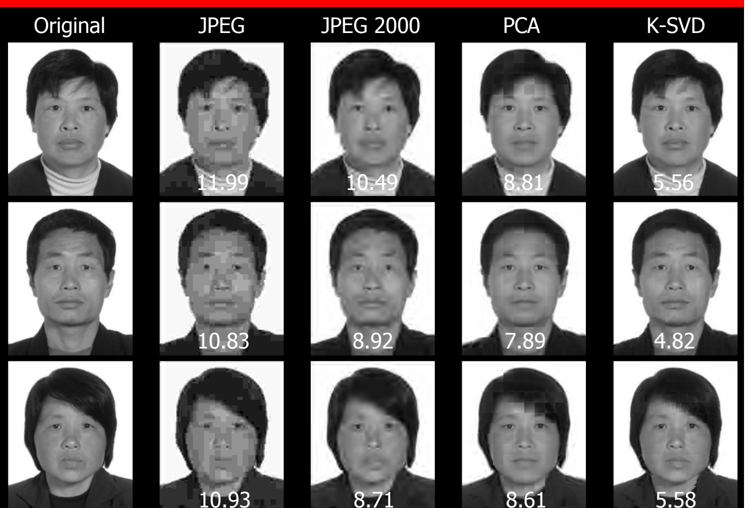
1. Detect features and align
2. Divide to disjoint patches, and sparse-code each patch
3. Quantize and entropy-code

Training set (2500 images)



Compression Results

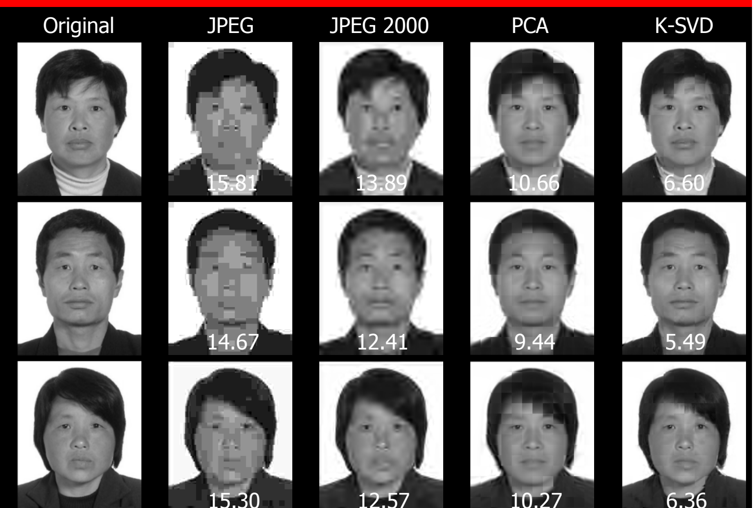
Results for 820 bytes per image



Bottom: RMSE values

Compression Results

Results for 550 bytes per image



Bottom: RMSE values

Today We Have Discussed

1. A Visit to *Sparseland*
   Introducing sparsity & overcompleteness
2. Transforms & Regularizations
   How & why should this work?
3. What about the dictionary?
   The quest for the origin of signals
4. Putting it all together
   Image filling, denoising, compression, …

Summary

Sparsity and over-completeness are important ideas for designing better tools in signal and image processing

Coping with an NP-hard problem

Approximation algorithms can be used, are theoretically established and work well in practice

What dictionary to use?

Several dictionaries already exist. We have shown how to practically train **D** using the K-SVD

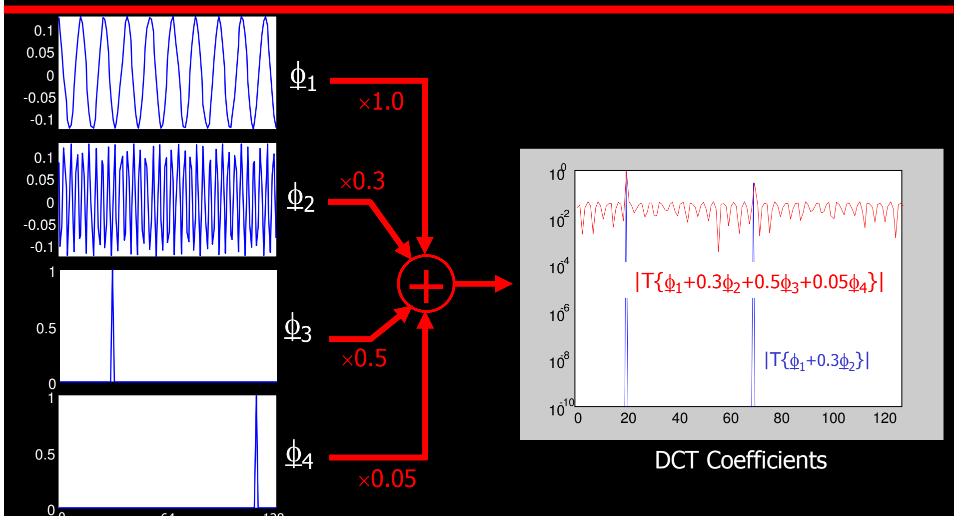
How is all this used?

We have seen inpainting, denoising and compression algorithms.

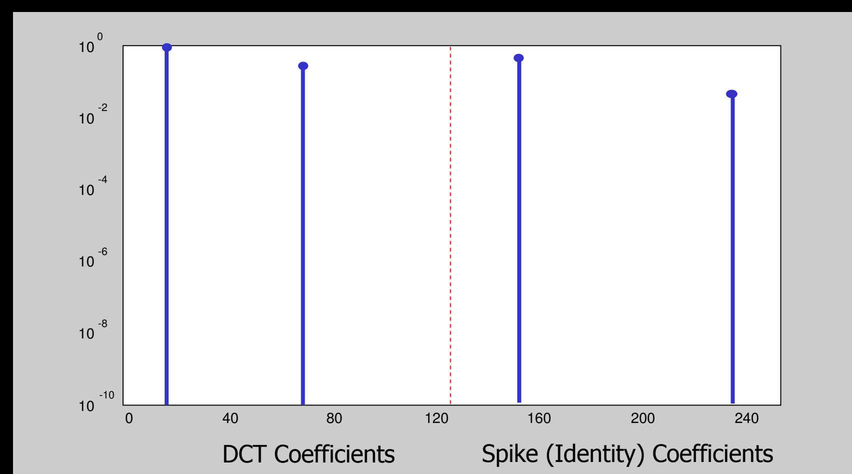
What next?

(a) Generalizations: multiscale, non-negative,…
(b) Speed-ups and improved algorithms
(c) Deploy to other applications

Why Over-Completeness?

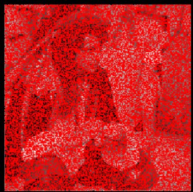


Desired Decomposition



Inpainting Results

70% Missing Samples

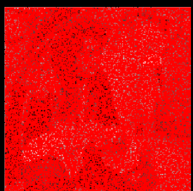


DCT (RMSE=0.04)

Haar (RMSE=0.045)

K-SVD (RMSE=0.03)

90% Missing Samples



DCT (RMSE=0.085_

Haar (RMSE=0.07)

K-SVD (RMSE=0.06)