

A Study of the K-SVD Algorithm for Designing Overcomplete Dictionaries for Sparse Representation

Lijun Xu

School of Mathematical Sciences
Dalian University of Technology

Supervised by

Bo Yu (DUT)

Yin Zhang (Rice University)

March 6, 2012

Introduction

This talk is about an effective method of training overcomplete dictionaries for sparse signal representation proposed in the following two papers.

1. M. Aharon, M. Elad, and A. Bruckstein. The K-SVD: An algorithm for designing of overcomplete dictionaries for sparse representation, IEEE Trans. on Signal Processing, vol. 54, no. 11, pp. 4311-4322, 2006.
2. R. Rubinstein, M. Zibulaevsky, and M. Elad. Efficient implementation of the k-svd algorithm using batch orthogonal matching pursuit. Technical Report CS-2008-08, Technion - Israel Institute of Technology, 2008

Introduction

- Complete:

a system $\{y_i\}$ in a Banach Space X is complete

if every element in X can be approximated arbitrarily well in norm by linear combinations of elements in $\{y_i\}$.

- Overcomplete:

if removal of an element from the system $\{y_i\}$ results in a complete system.

The arbitrary approximation in norm can be thought as a representation somehow.

Introduction

- In recent years there has been a growing interest in the study of sparse representation of signals which is based on an overcomplete dictionary.
- Applications that use sparse representation are many and include compression, regularization in inverse problems, feature extraction, and more.
- Sparsity in overcomplete dictionaries is the basis for a wide variety of highly effective signal and image processing techniques.

Sparse Representation

Model: Given a dictionary D and a signal y ,

$$\min_x \|x\|_0 \quad \text{s.t.} \quad y = Dx \quad \text{or}$$

$$\min_x \|x\|_0 \quad \text{s.t.} \quad \|y - Dx\|_2 \leq \varepsilon$$

- $D \in R^{n \times K}$: an overcomplete dictionary matrix,
- $y \in R^n$: a signal can be represented as a sparse linear combination of columns of D ,
- $x \in R^K$: a vector contains the representation coefficients of the signal y

How to choose D ?

The Choice of Dictionary

- Modeling Approach

Choosing a predesigned transform, such as

Wavelet [Mallat]

Curvelet [Candes & Donoho]

Contourlet [Do & Vetterli]

Bandlet [Mallat & Le-Pennec]

and others ...

Simpler, pre-specified, fast transform,

Their success depends on how suitable to signals.

How to adapt to other signals?

Bad for “smaller” signal families.

The Choice of Dictionary

- Training Approach

Train the dictionary directly based on the given examples, optimizing w.r.t. sparsity and other desired properties (normalized atoms, etc.).

Examples: ML (maximum likelihood)

MAP(maximum a-posteriori probability)

MOD (method of optimal direction)

ICA-like (independent component analysis)

and others ...

- Fits the data,
- Design fits the true objectives,
- Can be adapted to small families.
- No fast version,
- Slow training,

K-SVD Algorithm

[Aharon, Elad, & Bruckstein,(2004)]

- model:

$$\min_{D,X} \|Y - DX\|_F^2 \quad \text{subject to} \quad \forall i, \|x_i\|_0 \leq S$$

where

$Y = [y_1 \cdots y_P] \in R^{N \times P}$ represents P training signal set,(known)

$D \in R^{N \times K}$ is a designed dictionary matrix with K atoms,

$X \in R^{K \times P}$ is a sparse representations matrix with a sparsity level S

K-SVD Algorithm

- K-SVD Algorithm: two steps
 - (i) Sparse Coding:
Producing sparse representations matrix X ,
given the current dictionary D
 - (ii) Dictionary Update:(the main innovation)
Updating dictionary atoms, given the
current sparse representations

K-SVD: Sparse Coding Stage

- Fix D ,
- Aim to find the best coefficient matrix X ,

$$\min_X \|Y - DX\|_F^2 \quad s.t. \quad \forall i, \|x_i\|_0 \leq S$$



$$\min_{x_i} \|y_i - Dx_i\|_2^2 \quad s.t. \quad \forall i, \|x_i\|_0 \leq S, \quad i = 1, 2, \dots, P$$

(Ordinary Sparse Coding Problem)

- Use an approximate solving method.
 - Orthogonal Matching Pursuit (OMP)
 - Basis Pursuit (BP)
 - Focal Under-determined System Solver (FOCUSS)
 - And others...

K-SVD: Dictionary Update Stage

Goal: search for a better dictionary,

Scheme: update one column of D at a time,

—fixing all columns in D except one, d_k ,

—update it by optimizing the target function.

For the k^{th} atom (column of D),

$$\|Y - DX\|_F^2 = \left\| Y - \sum_{j=1}^K d_j g_j^T \right\|_F^2 = \left\| \left(Y - \sum_{j \neq k} d_j g_j^T \right) - d_k g_k^T \right\|_F^2 = \|E_k - d_k g_k^T\|_F^2$$

$$D = [d_1 \ d_2 \ \cdots \ d_K] \in R^{N \times K}, E_k = Y - \sum_{j \neq k} d_j g_j^T,$$

$$X = [x_1 \ \cdots \ x_P] = [g_1 \ g_2 \ \cdots \ g_K]^T \in R^{K \times P}$$

K-SVD: Dictionary Update Stage

- Update the k^{th} atom of D :

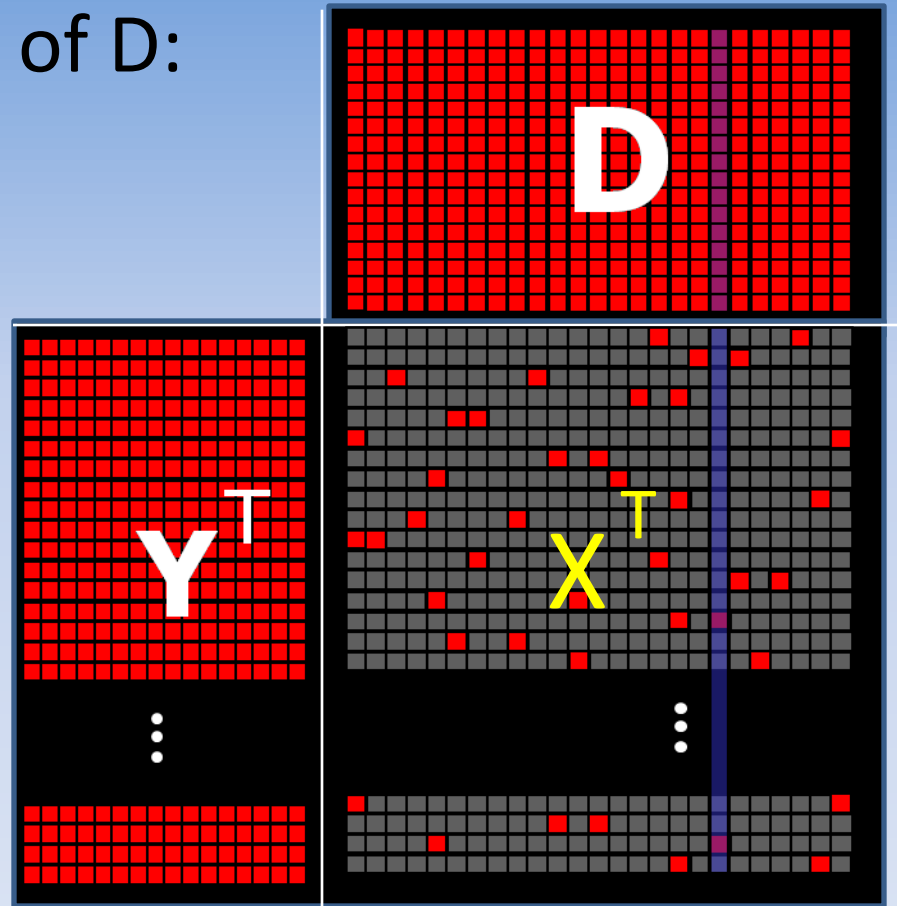
$$\min_{d_k} \|E_k - d_k g_k^T\|_F^2$$

We can do
better than
this

$$\min_{d_k, g_k} \|E_k - d_k g_k^T\|_F^2$$

$$E_k = Y - \sum_{j \neq k} d_j g_j^T \text{ (the residual)}$$

What about
sparsity ?



K-SVD: Dictionary Update Stage

To preserve sparsity, only consider those columns of E_k using d_k .

$$\min_{d_k, g_k} \left\| \begin{matrix} \text{green grid} & \dots & E_k & - & d_k & \dots & g_k^T & \text{red vector} \end{matrix} \right\|_F^2$$

Let I denote the indices of the signals in Y which use the k^{th} atom.
 $I = (i_1, \dots, i_k)$, $A_I = [a_{i_1} \cdots a_{i_k}]$ denote the submatrix of A indexed by I .

Update by optimizing:

$$\min_{d_k, \underline{g}_k} \|Y_I - DX_I\|_F^2 = \|\underline{E}_k - d_k \underline{g}_k^T\|_F^2$$

where

$\underline{E}_k = Y_I - \sum_{j \neq k} d_j X_{j,I}$ is the error matrix without the k^{th} atom.

$\underline{g}_k^T = X_{k,I}$ is the k^{th} row in X_I , namely the nonzeros of the k^{th} row in X .

K-SVD: Dictionary Update Stage

The simple rank-1 problem:

$$\min_{d_k, \underline{g}_k} \left\| \underline{E}_k - d_k \underline{g}_k^T \right\|_F^2 \quad \text{s.t.} \quad \|d_k\|_2 = 1,$$

Solve directly via SVD: $\underline{E}_k = U \Lambda V$

$$d_k = u_1, \underline{g}_k = \sigma_1 v_1,$$

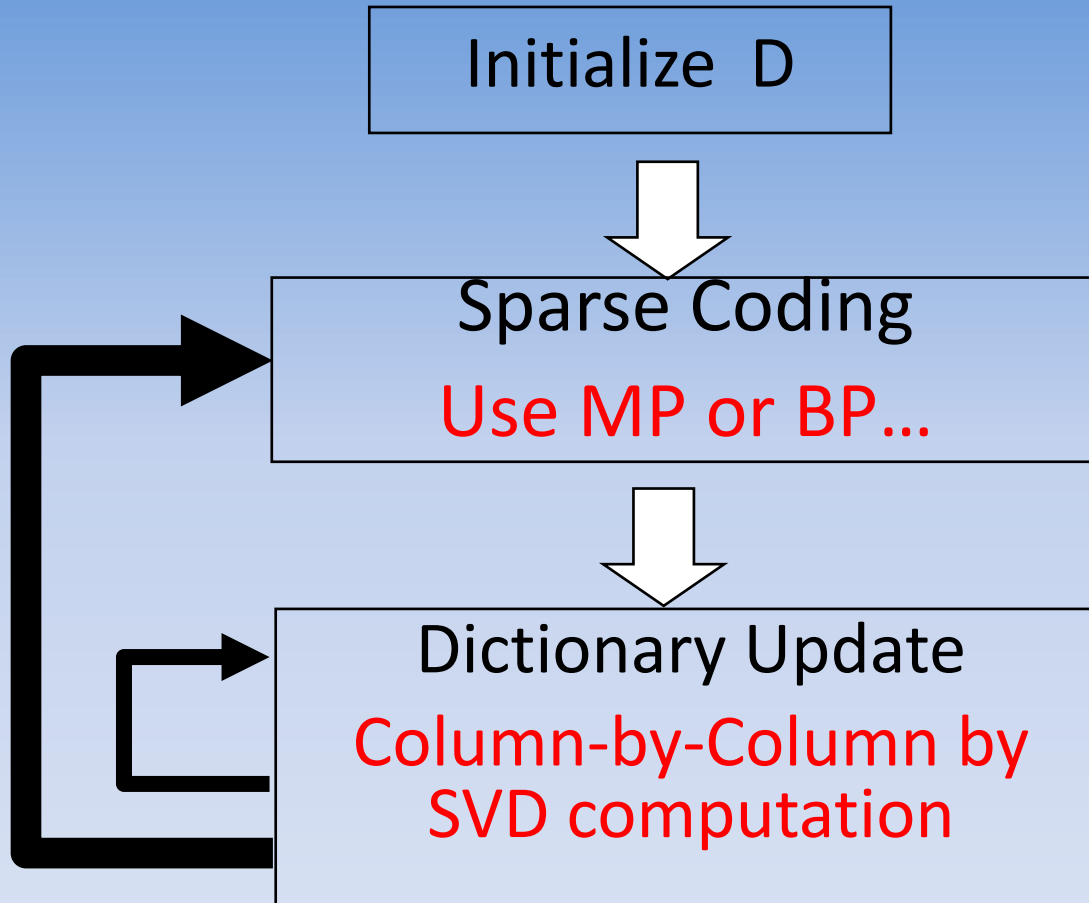
Update all K atoms of D column by column in the same fashion and get a new dictionary D^+ .

Then fix this D^+ and go to sparse coding stage,

.....

Untill reach stopping rule.

K-SVD: frame



Algorithm K-SVD^[2]

1: Input: Signal set Y , initial dictionary D_0 , target sparsity S , number of iteration L .

2: Output: Dictionary D and sparse matrix X such that $Y \approx DX$.

3: Init: Set $D := D_0$

4: **for** $n = 1, \dots, L$ **do**

5: $\forall i: x_i = \text{Arg min}_x \|y_i - Dx\|_2^2$ subject to $\|x\|_0 \leq S$  Sparse coding process

6: **for** $j = 1, \dots, K$ **do**

7: $d_j := 0$

8: $I := \{\text{indices of the signals in } Y \text{ whose representations use } d_j\}$

9: $E := Y_I - DX_I = Y_I - \sum_{l \neq j} d_l X_{l,I}$

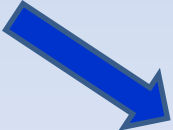
10: $\{d, g\} := \text{Arg min}_{d,g} \|E - dg^T\|_F^2$ subject to $\|d\|_2 = 1$

11: $d_j := d$

12: $X_{j,I} := g^T$

13: **end for**

14: **end for**



Dictionary update process

K-SVD: Summary

The K-SVD algorithm is an iterative method that alternates between sparse coding of the examples based on the current dictionary, and a process of updating the dictionary atoms to better fit the data.

It is flexible and can work with any pursuit method (e.g. BP, MP, or FOCUSS) which is used in sparse coding part. And the dictionary designed by the K-SVD performs well for both synthetic and real images in applications.

But the K-SVD algorithm is quite computationally demanding, especially when the dimensions of the dictionary increase or the number of training signals becomes large.

Approximate K-SVD:

Rubinstein, Zibulevsky & Elad (08)

In practice, the exact solution of

$$\{d, g\} := \operatorname{Arg} \min_{d, g} \|E - dg^T\|_F^2 \quad \text{subject to} \quad \|d\|_2 = 1$$

can be computationally difficult, as the size of E is proportional to the number of training signals.

Indeed, the goal of K-SVD is really to improve a given initial dictionary, not find an optimal one.

Therefore, a much quicker approach is to use an approximate solution just as long as it ensures a reduction of the final target function.

Approximate K-SVD:

As to the dictionary update step,

$$\{d, g\} := \underset{d, g}{\operatorname{Arg\,min}} \|E - dg^T\|_F^2 \quad \text{subject to} \quad \|d\|_2 = 1$$

using a single iteration of alternate-optimization over the atom d and the coefficients row g^T , which is given by

$$\begin{cases} \text{fix } g, \\ \min_d \|E - dg^T\|_F^2 \end{cases} \quad \text{s.t.} \quad \|d\|_2 = 1 \quad \Rightarrow \quad \boxed{d := Eg / \|Eg\|_2}$$

$$\begin{cases} \text{fix } d, \\ \min_g \|E - dg^T\|_F^2 \end{cases} \quad \Rightarrow \quad \boxed{g := E^T d}$$

Algorithm Approximate K-SVD:

1: Input: Signal set Y , initial dictionary D_0 , target sparsity S , number of iteration L .

2: Output: Dictionary D and sparse matrix X such that $Y \approx DX$.

3: Init: Set $D := D_0$

4: **for** $n = 1, \dots, L$ **do**

5: $\forall i: x_i = \text{Arg min}_x \|y_i - Dx\|_2^2$ subject to $\|x\|_0 \leq S$

6: **for** $j = 1, \dots, K$ **do**

7: $d_j := 0$

8: $I := \{\text{indices of the signals in } Y \text{ whose representations use } d_j\}$

9: $g := X_{j,I}^T$

10: $d := Y_I g - DX_I g$

11: $d := d / \|d\|_2$

12: $g := Y_I^T d - (DX_I)^T d$

13: $d_j := d$

14: $X_{j,I} := g^T$

15: **end for**

16: **end for**

$$9: \quad \underline{E := Y_I - DX_I = Y_I - \sum_{l \neq j} d_l X_{l,I}}$$

$$10: \quad \underline{\{d, g\} := \text{Arg min}_{d,g} \|E - dg^T\|_F^2} \quad \text{subject to} \quad \|d\|_2 = 1$$

$$11: \quad d_j := d$$

$$12: \quad X_{j,I} := g^T$$

K-SVD algorithm

Approximate K-SVD:

- Eliminate the need to explicitly compute the matrix E.
- Time and memory saving.
- Actually it's faster and more efficient using Batch-OMP instead of OMP in the sparse coding step for large signals.
- Complexity in one iteration

Assuming $S \ll K \sim N \ll P$.

$$T_{K-SVD} \approx P(S^2 K + 2NK)$$

↑
sparsity

↗
the number of atoms of dictionary

↗
dimension of a signal

↘
the number of training signals

K-SVD Algorithm

[Aharon, Elad, & Bruckstein,(2004)]

- model:

$$\min_{D,X} \|Y - DX\|_F^2 \quad \text{subject to} \quad \forall i, \|x_i\|_0 \leq S$$

where

$Y = [y_1 \cdots y_P] \in R^{N \times P}$ represents P training signal set,(known)

$D \in R^{N \times K}$ is a designed dictionary matrix with K atoms,

$X \in R^{K \times P}$ is a sparse representations matrix with a sparsity level S

K-SVD Algorithm

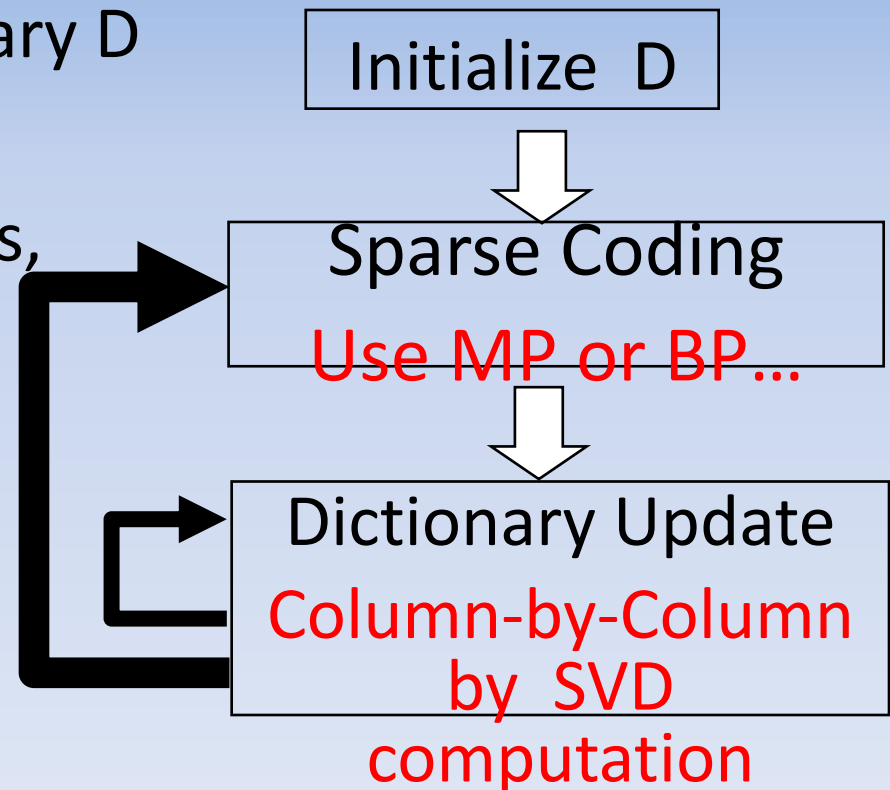
- K-SVD Algorithm: two steps

(i) Sparse Coding:

Producing sparse representations matrix X ,
given the current dictionary D

(ii) Dictionary Update:

Updating dictionary atoms,
given the current sparse
representations



K-SVD: Sparse Coding Stage

- Fix D ,
- Aim to find the best coefficient matrix X ,

$$\min_X \|Y - DX\|_F^2 \quad s.t. \quad \forall i, \|x_i\|_0 \leq S$$



$$\min_{x_i} \|y_i - Dx_i\|_2^2 \quad s.t. \quad \forall i, \|x_i\|_0 \leq S, \quad i = 1, 2, \dots, P$$

(Ordinary Sparse Coding Problem)

- Use an approximate solving method.
 - Orthogonal Matching Pursuit (OMP)
 - Basis Pursuit (BP)
 - Focal Under-determined System Solver (FOCUSS)
 - And others...

K-SVD: Dictionary Update Stage

Goal: search for a better dictionary,

Scheme: update one column of D at a time,

—fixing all columns in D except one, d_k ,

—update it by optimizing the target function.

For the k^{th} atom (column of D),

$$\|Y - DX\|_F^2 = \left\| Y - \sum_{j=1}^K d_j g_j^T \right\|_F^2 = \left\| \left(Y - \sum_{j \neq k} d_j g_j^T \right) - d_k g_k^T \right\|_F^2 = \|E_k - d_k g_k^T\|_F^2$$

$$D = [d_1 \ d_2 \ \cdots \ d_K] \in R^{N \times K}, E_k = Y - \sum_{j \neq k} d_j g_j^T,$$

$$X = [x_1 \ \cdots \ x_P] = [g_1 \ g_2 \ \cdots \ g_K]^T \in R^{K \times P}$$

K-SVD: Dictionary Update Stage

- Update the k^{th} atom of D :

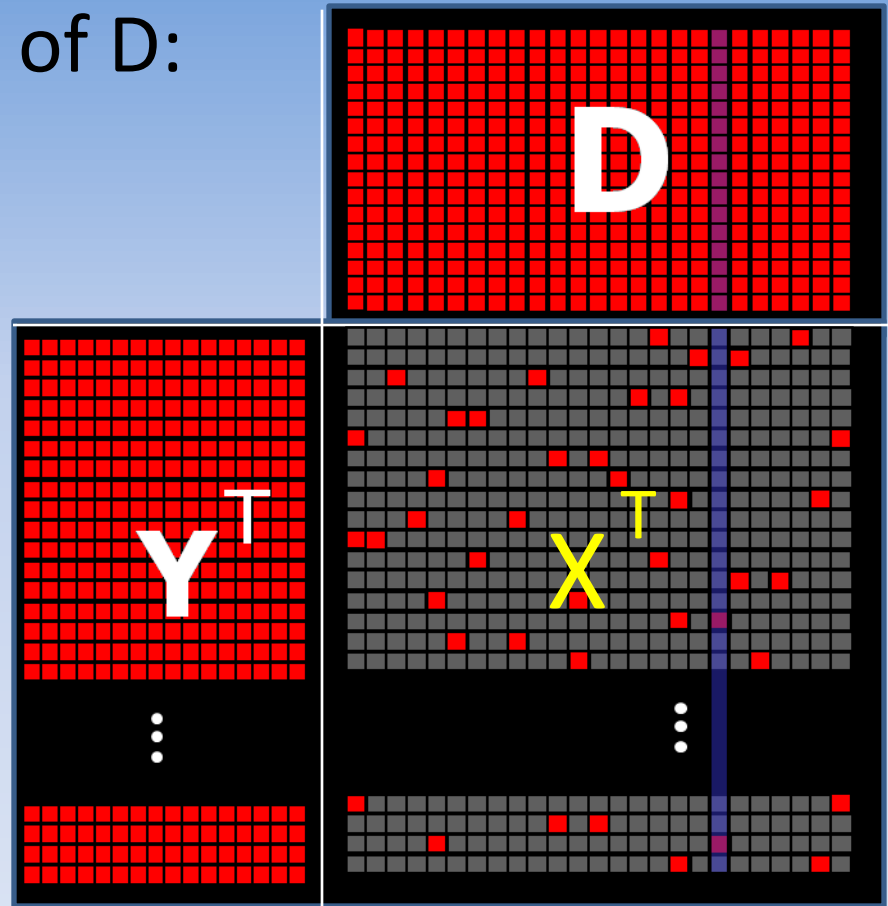
$$\min_{d_k} \|E_k - d_k g_k^T\|_F^2$$

We can do
better than
this

$$\min_{d_k, g_k} \|E_k - d_k g_k^T\|_F^2$$

$$E_k = Y - \sum_{j \neq k} d_j g_j^T \text{ (the residual)}$$

What about
sparsity ?



K-SVD: Dictionary Update Stage

To preserve sparsity, only consider those columns of E_k using d_k .

$$\min_{d_k, g_k} \left\| \begin{matrix} \text{green grid} & \dots & E_k & - & d_k & \dots & g_k^T & \text{red grid} \end{matrix} \right\|_F^2$$

Let I denote the indices of the signals in Y which use the k^{th} atom.
 $I = (i_1, \dots, i_k)$, $A_I = [a_{i_1} \cdots a_{i_k}]$ denote the submatrix of A indexed by I .

Update by optimizing:

$$\min_{d_k, \underline{g}_k} \|Y_I - DX_I\|_F^2 = \|\underline{E}_k - d_k \underline{g}_k^T\|_F^2$$

where

$\underline{E}_k = Y_I - \sum_{j \neq k} d_j X_{j,I}$ is the error matrix without the k^{th} atom.

$\underline{g}_k^T = X_{k,I}$ is the k^{th} row in X_I , namely the nonzeros of the k^{th} row in X .

K-SVD: Dictionary Update Stage

The simple rank-1 problem:

$$\min_{\underline{d}_k, \underline{g}_k} \left\| \underline{E}_k - \underline{d}_k \underline{g}_k^T \right\|_F^2 \quad \text{s.t.} \quad \left\| \underline{d}_k \right\|_2 = 1,$$

Solve directly via SVD ($\underline{d}_k = u_1, \underline{g}_k = \sigma_1 v_1$),
or approximate methods.

Update all K atoms of D column by column in the same fashion and get a new dictionary D^+ .

Then fix this D^+ and go to sparse coding stage,

.....

Until reach stopping rule.

K-SVD: A Synthetic Experiment

- Generation of the data to train on:

Create a 20*50 random matrix D (normalized),

Produce 1500 data signals $\{y_i\}_{i=1}^{1500}$,

each created by a linear combination of 3 different columns of D and adding white Gaussian noise with varying SNR (Signal to Noise Ratio)

- Applying the K-SVD to solve :

$$\min_{D,X} \|Y - DX\|_F^2 \quad \text{s.t.} \forall i, \|x_i\|_0 \leq S$$

initial dictionary D_0 (initialized with sigals),

sparsity $S=3$, number of iteration $L=80$.

using OMP in sparse coding step.

K-SVD: A Synthetic Experiment

- Compare the result \underline{D} with the known D :
For every column d_i of D , find the closest column d_j in \underline{D} measuring via $1 - |d_i^T \underline{d}_j|$.
If $1 - |d_i^T \underline{d}_j| < 0.01$, it was considered a success.
- Repeat 50 times trials for a same noise level:
Compute the number of successes in each trial.
- Results:
Compare with other algorithm:
MOD(Method of Optimal Directions) and
MAP-based(Maximum A-posteriori Probability)

K-SVD: A Synthetic Experiment

The results for noise levels of 10dB, 20dB, 30dB and no noise:

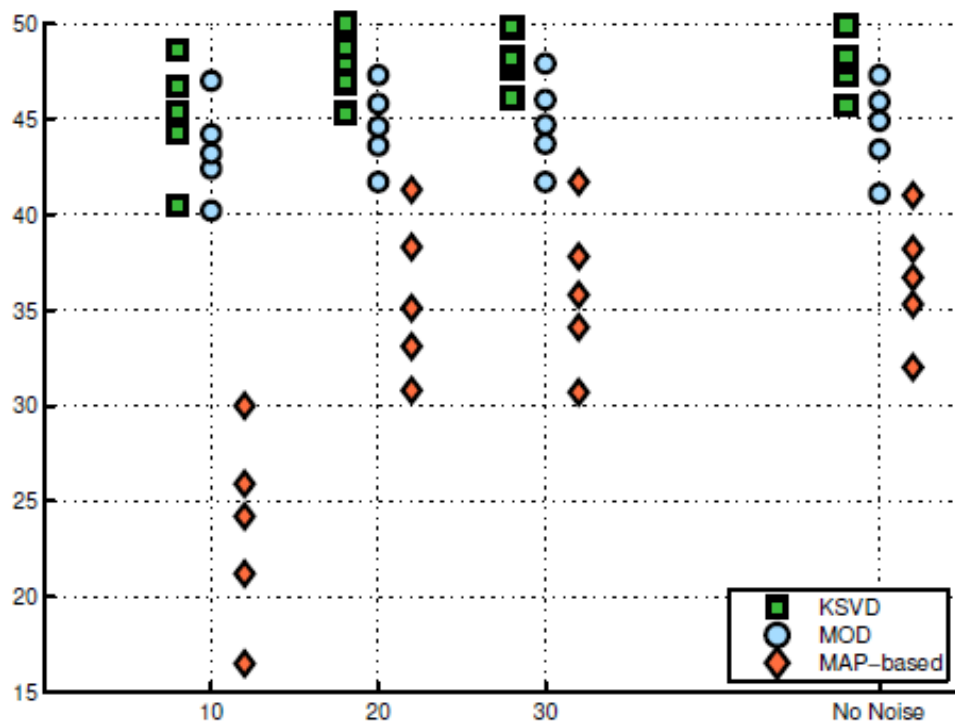


Fig. 3. Synthetic results: for each of the tested algorithms and for each noise level, 50 trials were conducted, and their results sorted. The graph labels represent the mean number of detected atoms (out of 50) over the ordered tests in groups of 10 experiment.

K-SVD on Image:

- Training Data:

11,000 examples of 8×8 block patches taken from a database of face images in various location. A random collection of 500 blocks is presented in Figure 4.

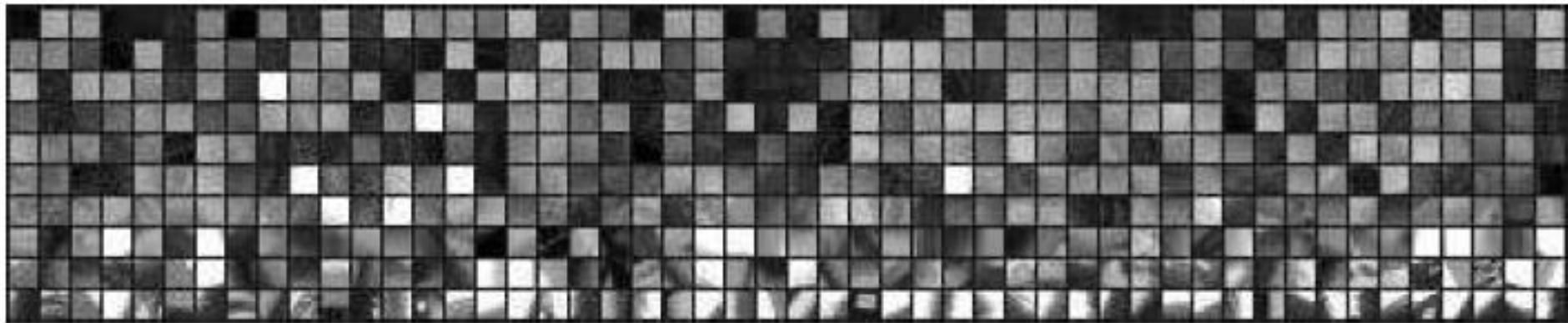



Fig. 4. A collection of 500 random blocks that were used for training, sorted by their variance.

K-SVD on Image:

- Every 8×8 block represents a column of 64×11000 matrix Y . For a block patch of size 8×8 pixel,
 $\longrightarrow [y_{i1} \ y_{i2} \ \dots \ y_{i64}]^T$
- Running the K-SVD:
size of D : 64×441
sparsity of X : 10
sparse coding algorithm: OMP
- Comparison Dictionaries:
with the overcomplete Haar dictionary and overcomplete DCT dictionary

K-SVD on Image:

- Dictionary size : 64×441 .
every column presents a block patch of size 8×8 pixels

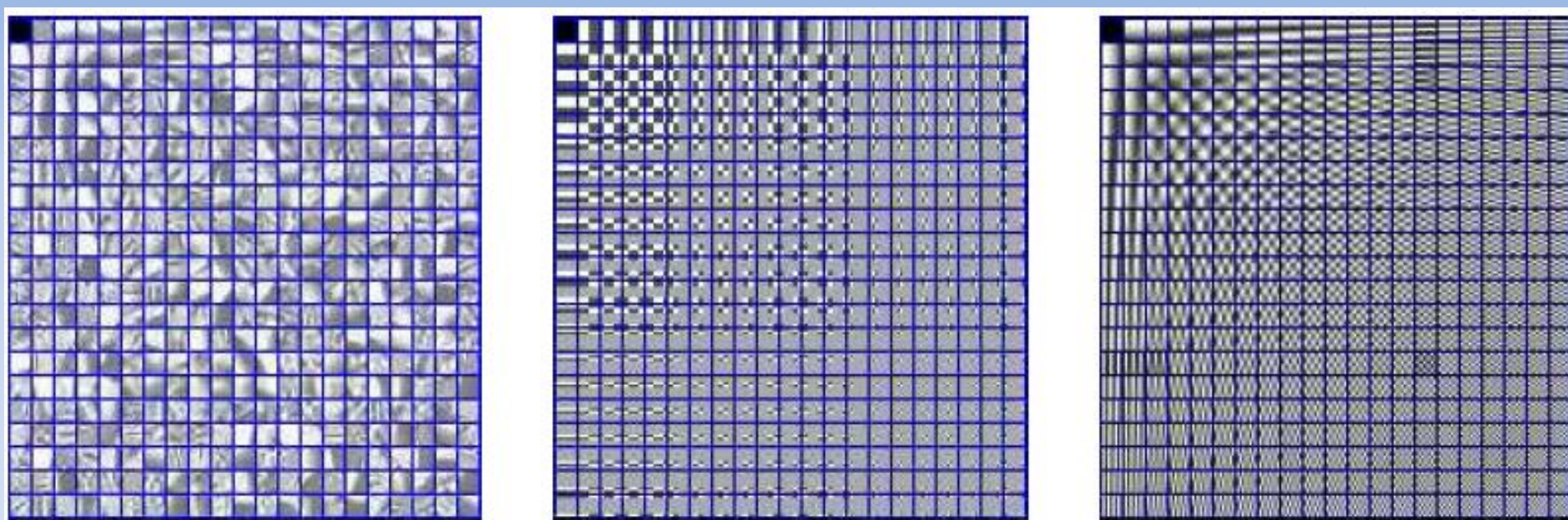


Fig. 5. The learned dictionary (left). Its elements are sorted in an ascending order of their variance, and stretched to maximal range for display purposes. The overcomplete separable Haar dictionary (middle), and the overcomplete DCT dictionary (right) are used for comparison.

K-SVD on Image:

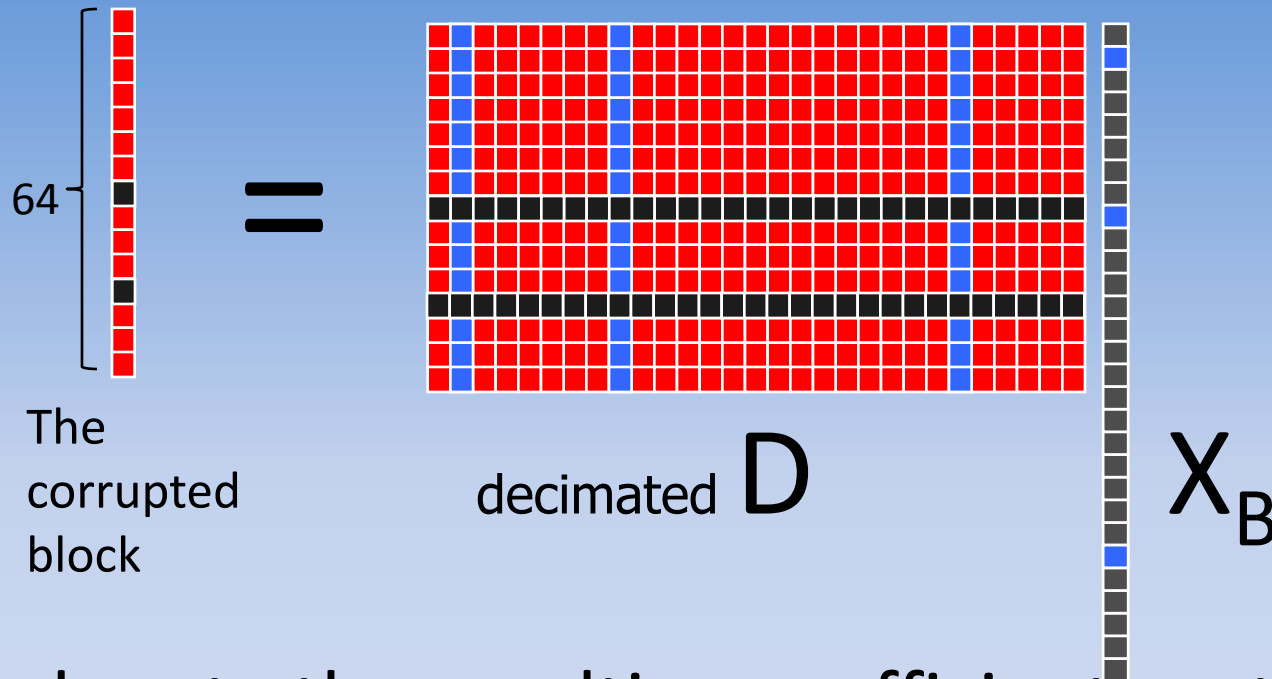
- Application: Filling-In Missing Pixels

Test image: one random full face image consisting of 594 blocks
(non of them were used for training)

For each 8×8 block B:

- 1) Delete a fraction r of the pixels in random location (set to zero)
- 2) Using OMP to the corrupted block under the learned D , the overcomplete Haar dictionary and the overcomplete DCT dictionary.
 - ◆ actually using a decimated dictionary with rows removed

K-SVD on Image:



denote the resulting coefficient vector as X_B

3) The reconstructed block $B' = D \bullet X_B$,

4) The reconstructed error: $\sqrt{\|B - B'\|_F^2 / 64}$

K-SVD on Image:

- Compute the mean reconstruction errors for all blocks and all corruption rates , displayed in Figure 6

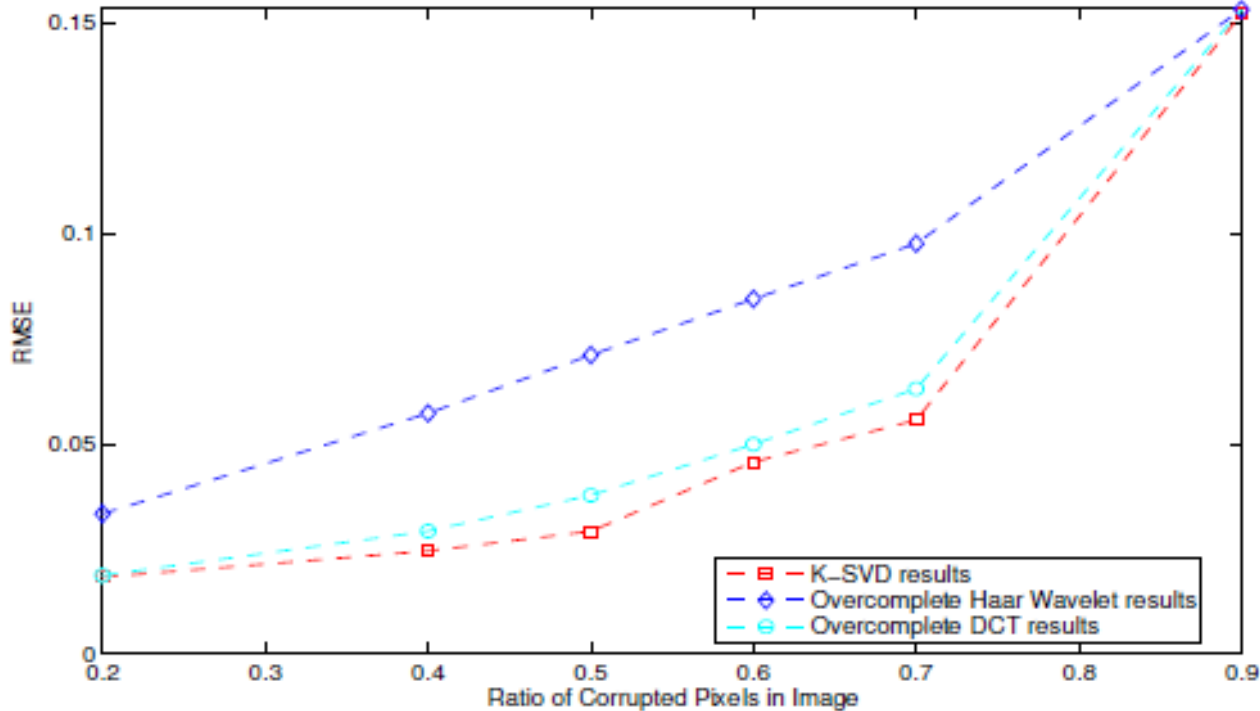


Fig. 6. The RMSE for 594 new blocks with missing pixels using the learned dictionary, overcomplete Haar dictionary, and overcomplete DCT dictionary.

K-SVD on Image:

- Two corrupted images and their reconstructions

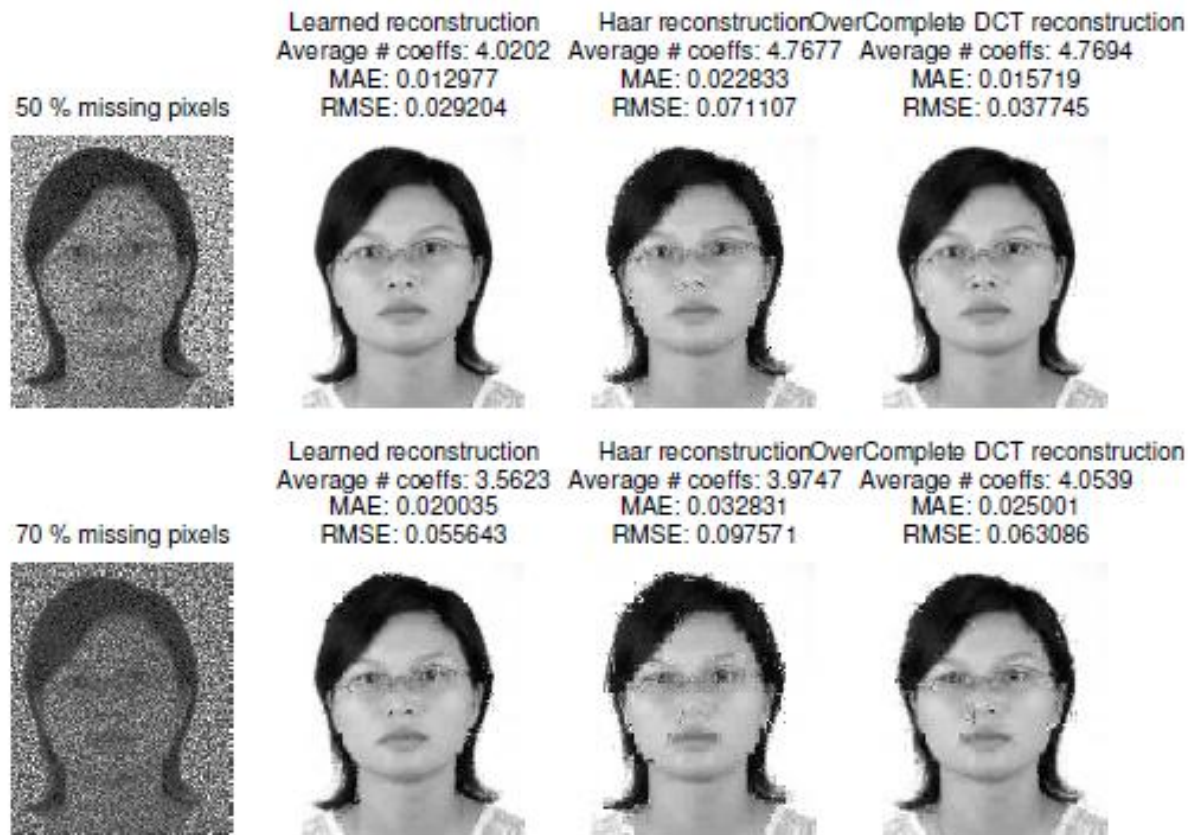


Fig. 7. The corrupted image (left) with the missing pixels marked as points, and the reconstructed results by the learned dictionary, the overcomplete Haar dictionary, and the overcomplete DCT dictionary, respectively. The different rows are for 50% and 70% of missing pixels.

Some experiments on K-SVD & SeMF

Review:

$$\min_{D, X} \|Y - DX\|_F^2 \quad \text{subject to} \quad \forall i, \|x_i\|_0 \leq S$$

K-SVD Algorithm: (two steps)

- sparse coding (OMP)
given Y and D , producing a X with sparsity S .
- dictionary updating
updating D columns by columns by using SVD or approximate approaches.

Introduction of ADM :

- Original model

$$\min_{W,H} \|A - WH^T\|_F^2 \quad \text{s.t.} \quad W \in \mathbb{T}_1, H \in \mathbb{T}_2$$

- Model with splitting variables:

$$\min_{W,H,S_1,S_2} \|A - WH^T\|_F^2 \quad \text{s.t.} \quad W = S_1 \in \mathbb{T}_1, H = S_2 \in \mathbb{T}_2$$

- ♦ Splitting variable S_1 separates W from \mathbb{T}_1 (similarly for S_2).
- ♦ Separations facilitates alternating direction methods.

Introduction of ADM :

- Alternating Direction Method Framework

Augmented Lagrangian (AL):

$$\mathcal{L}_{\mathcal{A}}(W, H, S, \Lambda) = \|A - WH^T\|_F^2 + \beta_1 \|W - S_1\|_F^2 + \beta_2 \|H - S_2\|_F^2 - 2\Lambda_1 \bullet (W - S_1) - 2\Lambda_2 \bullet (H - S_2)$$

Alternately minimizing it over one variable:

$$\text{ALADM : } \begin{cases} W^{k+1} \leftarrow \arg \min_W \mathcal{L}_{\mathcal{A}}(W, H^k, S^k, \Lambda^k) \\ H^{k+1} \leftarrow \arg \min_H \mathcal{L}_{\mathcal{A}}(W^k, H, S^k, \Lambda^k) \\ S_1^{k+1} \leftarrow \text{Proj}_{\mathbf{T}_1}(W^{k+1} - \Lambda_1^k / \beta_1) \\ S_2^{k+1} \leftarrow \text{Proj}_{\mathbf{T}_2}(H^{k+1} - \Lambda_2^k / \beta_2) \\ \Lambda_1^{k+1} \leftarrow \Lambda_1^k - \beta_1(W^{k+1} - S_1^{k+1}) \\ \Lambda_2^{k+1} \leftarrow \Lambda_2^k - \beta_2(H^{k+1} - S_2^{k+1}) \end{cases}$$

Some experiments on K-SVD & SeMF

Structure-enforcing Matrix Factorization (SeMF) :

- Splitting variables

$$\min_{D, X} \|Y - DX\|_F^2 \quad \text{subject to} \quad D = S_1 \in T_1, X = S_2 \in T_2$$

- ADM for Augmented Lagrangian (AL):

$$\begin{aligned} L_Y(D, X, S, \Lambda) = & \|Y - DX\|_F^2 + \beta_1 \|D - S_1\|_F^2 + \beta_2 \|X - S_2\|_F^2 \\ & - 2\Lambda_1 \bullet (D - S_1) - 2\Lambda_2 \bullet (X - S_2) \end{aligned}$$

Notes: T_1 (normalized), T_2 (sparse) allow easy projections.

Some experiments on K-SVD & SeMF

1. Synthetic experiments

D: random 20×50 matrix with $\|d_i\|_2 = 1$

X: sparse 50×1500 matrix, sparsity is 3 with random location and value,

Y: $D \times X + \text{noise}$, $\{y_i\}_{i=1}^{1500}$

Solving $\min_{D, X} \|Y - DX\|_F^2$ s.t. $\forall i, \|x_i\|_0 \leq 3$ with SeMF and K-SVD,

Denote the learning dictionary as **D**,

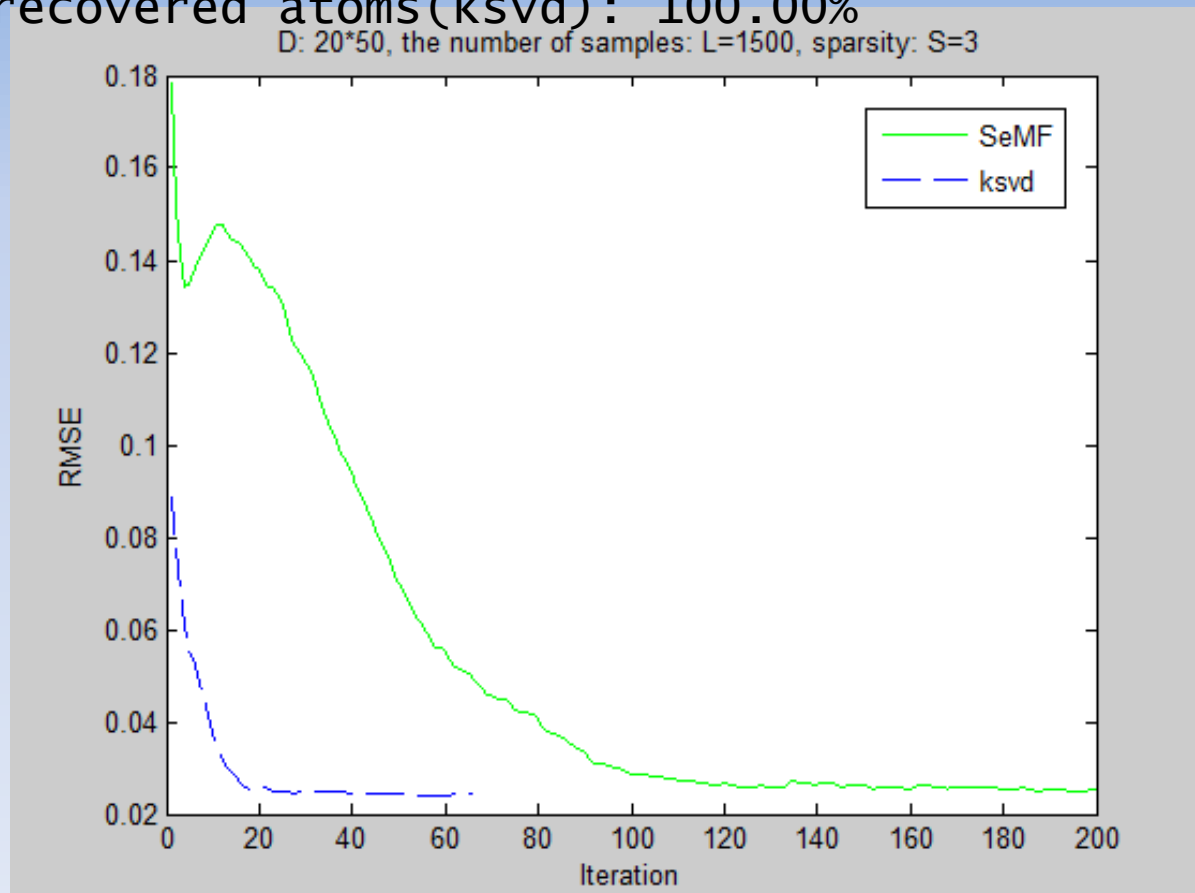
Measure the distance via $1 - |d_i^T \underline{d}_j|$

If $1 - |d_i^T \underline{d}_j| < 0.01$, we consider as a success.

Compute the number of the successful columns.

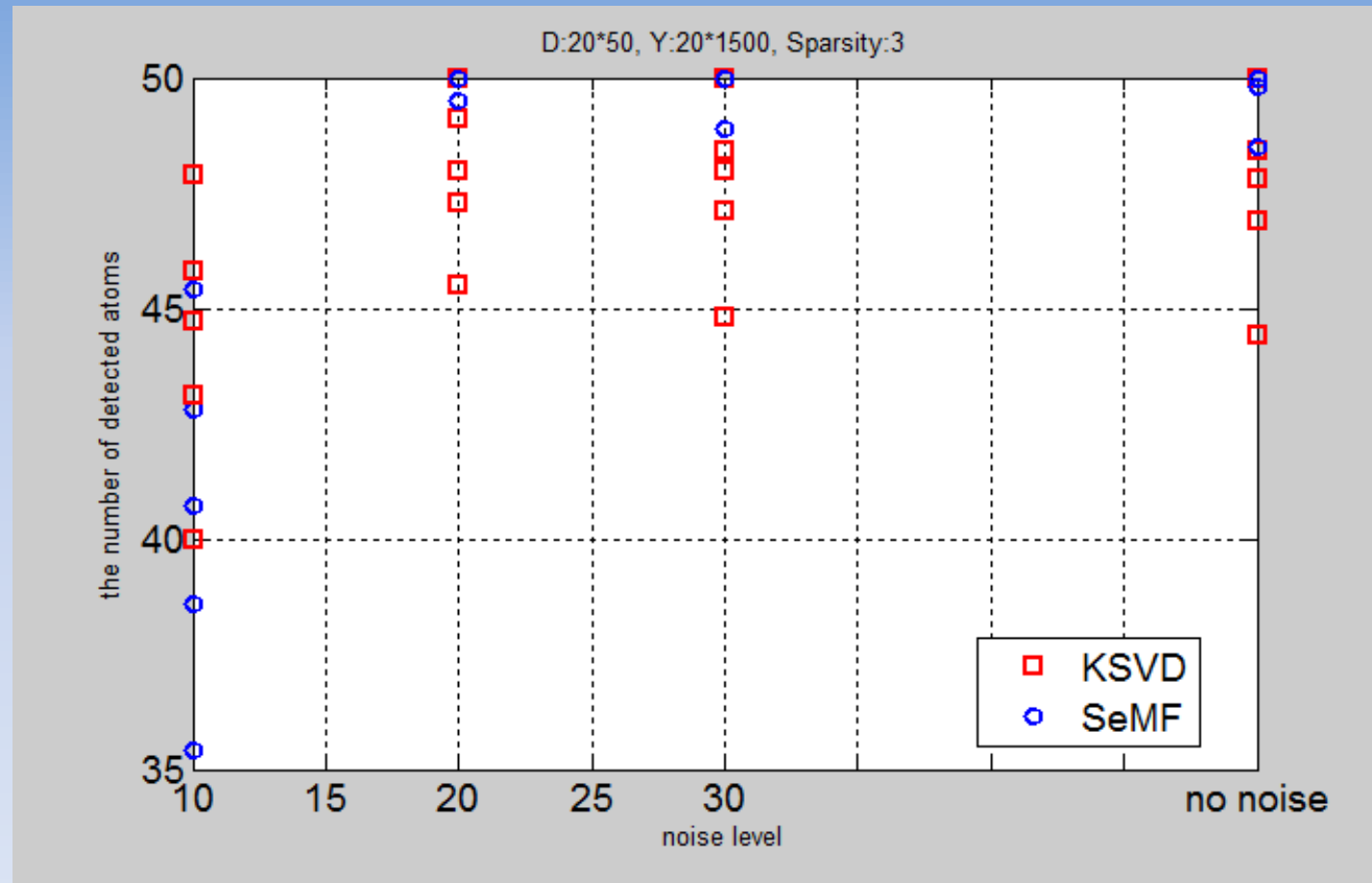
Some experiments on K-SVD & SeMF

Dictionary size: 20 x 50 Sparsity: 3 Number of examples: 1500
elapsed time (SeMF): 3.08729 (s) iteration(SeMF): 200
elapsed time (ksvd): 3.98188 (s) iteration(ksvd): 66
Ratio of recovered atoms(SeMF): 100.00%
Ratio of recovered atoms(ksvd): 100.00%



Some experiments on K-SVD & SeMF

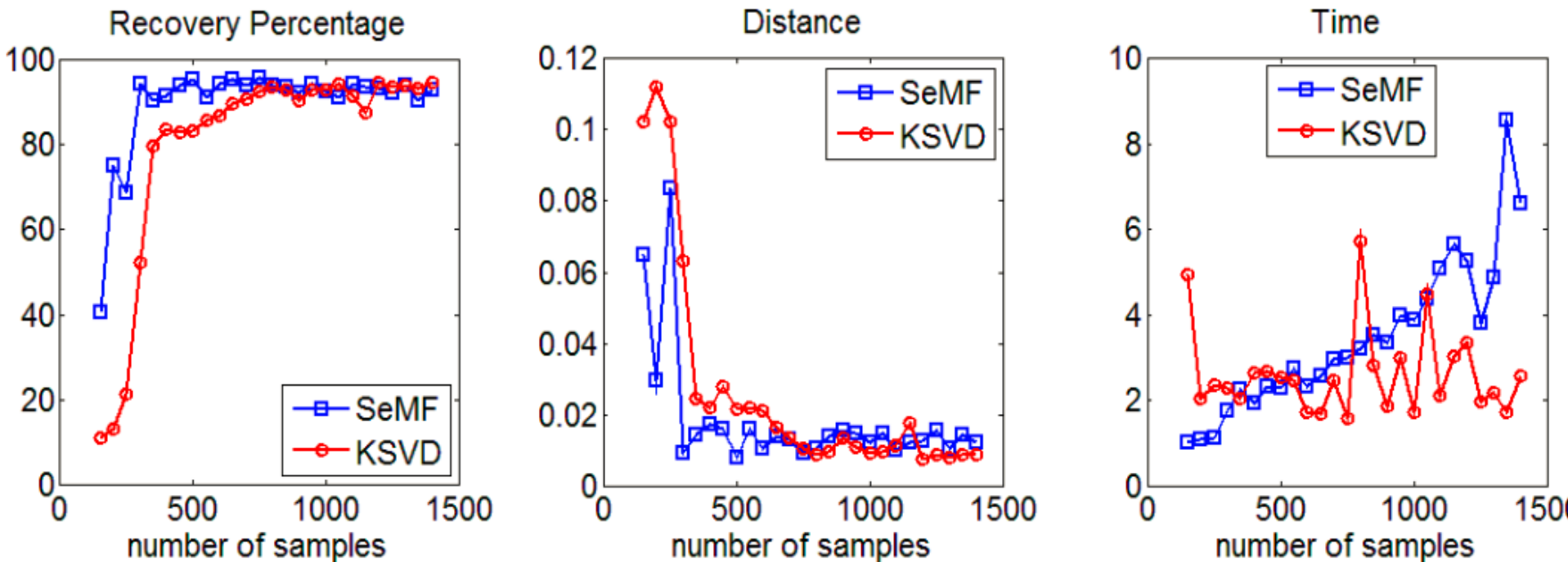
➤ The results for noise levels of 10dB, 20dB, 30dB and no noise:



Repeat 50 times trials for every level, the mean number of detected columns in groups of 10 experiments.

Some experiments on K-SVD & SeMF

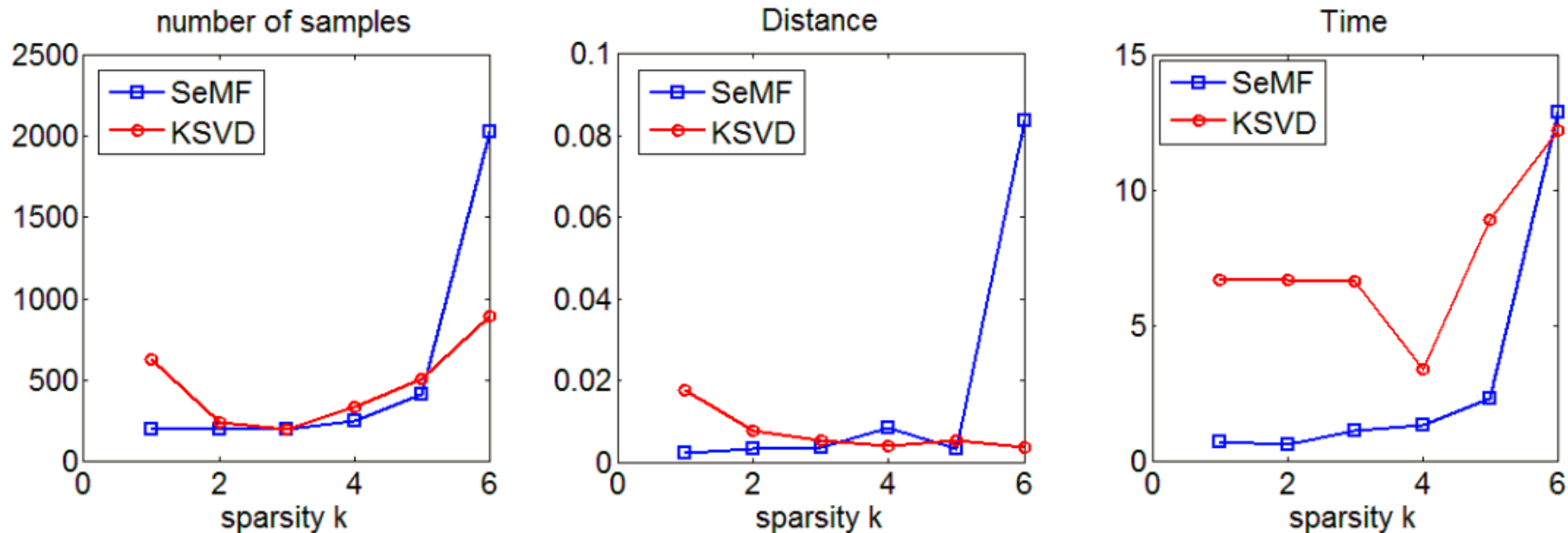
- The results for different number of samples:
D:20*50, sparsity: 3, the number of samples: L=150:50:1400



Some experiments on K-SVD & SeMF

- Number of samples to achieve 95% recovery with different sparsity level.

D:20*50, sparsity: $k=1:6$, number of samples: $L=200:10:2000$.
repeat 5 times for each k and make an average L .



Some experiments on K-SVD & SeMF

2. Applications on denoising

a) Given a noisy image y , we can clean it by solving (OMP)

$$\hat{\alpha} = \underset{\alpha}{\operatorname{ArgMin}} \|y - D\alpha\|_F^2 \quad s.t. \quad \|\alpha\|_0 \leq k \quad \longrightarrow \quad \hat{x} = D\hat{\alpha}$$

b) Need a dictionary D . (K-SVD & SeMF)

c) Training data Y :

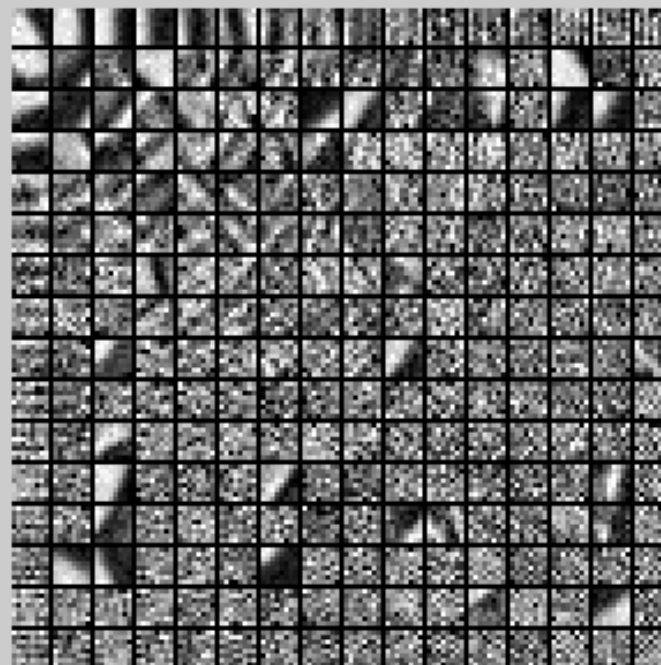
- ☐ Use the **corrupted image** itself !
- ☐ Simply sweep through all $N \times N$ patches (with overlaps) and use them to train
- ☐ Image of size 1000×1000 pixels
~ 10^6 examples to use – more than enough.

Using this Y , compute a dictionary D by K-SVD & SeMF, respectively.

Original image



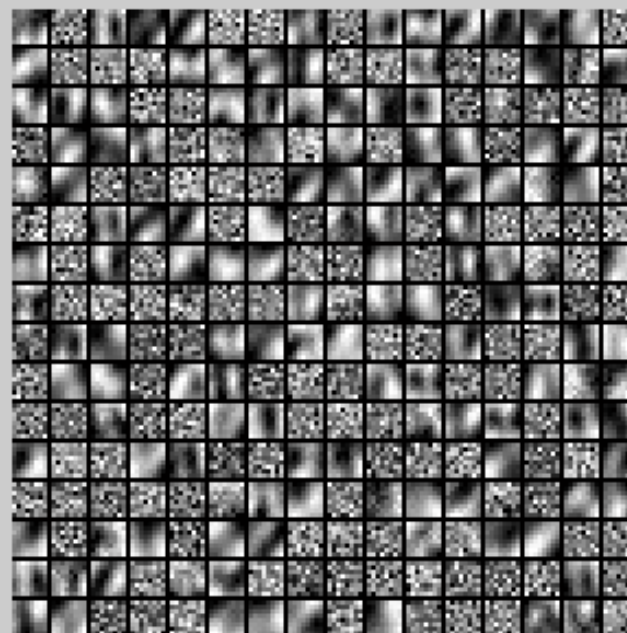
KSVD Trained dictionary



Noisy image, PSNR = 22.12dB



SeMF Trained dictionary, penalty= 1.376e+002



Some experiments on K-SVD & SeMF

SeMF Denoised image, PSNR = 31.69dB



KSVD Denoised image, PSNR: 32.02dB



Reference:

1. M. Aharon, M. Elad, and A. Bruckstein. The K-SVD: An algorithm for designing of overcomplete dictionaries for sparse representation, IEEE Trans. on Signal Processing, vol. 54, no. 11, pp. 4311-4322, 2006.
2. R. Rubinstein, M. Zibulaevsky, and M. Elad. Efficient implementation of the k-svd algorithm using batch orthogonal matching pursuit. Technical Report CS-2008-08, Technion - Israel Institute of Technology, 2008
3. J.A. Tropp, A.C. Gilbert. Signal recovery from random measurements via orthogonal matching pursuit. IEEE Trans. Inform. Theory, 53 (12) (2007), pp. 4655–4666