Machine Learning for Signal Processing
Sparse and Overcomplete Representations

Bhiksha Raj
(slides from Sourish Chaudhuri)
Oct 18, 2016
Key Topics in this Lecture

• Basics – Component-based representations
  – Overcomplete and Sparse Representations,
  – Dictionaries

• Pursuit Algorithms

• How to learn a dictionary

• Why is an overcomplete representation powerful?
Representing Data

Dictionary (codebook)
Representing Data

Dictionary

Atoms

Sparse and Overcomplete Representations
Representing Data

Dictionary

Atoms

Each atom is a basic unit that can be used to “compose” larger units.
Representing Data

Atoms

Sparse and Overcomplete Representations
Representing Data

Atoms

Sparse and Overcomplete Representations
Representing Data

Atoms

Many such bases (concepts)
Representing Data
Representing Data

Using concepts that we know...
Representing Data

Using concepts that we know...
Representing Data

Sparse and Overcomplete Representations
Representing Data

Sparse and Overcomplete Representations
Representing Data

Sparse and Overcomplete Representations
Representing Data

Sparse and Overcomplete Representations
Representing Data

\[ \mathbf{D} \alpha = \mathbf{X} \]

Sparse and Overcomplete Representations
Representing Data

\[ \text{Unknown} = \alpha X \]

Sparse and Overcomplete Representations
Quick Linear Algebra Refresher

• Remember, #(Basis Vectors) = #unknowns

\[ D \alpha = X \]

- Basis Vectors (from Dictionary)
- Weights
- Input data

Sparse and Overcomplete Representations
Overcomplete Representations

• What is the dimensionality of the input image? (say 64x64 image)

  ➢ 4096

• What is the dimensionality of the dictionary? (each image = 64x64 pixels)

  ➢ 4096 x N
Overcomplete Representations

• What is the dimensionality of the input image? (say 64x64 image)
  ➢ 4096

• What is the dimensionality of the dictionary? (each image = 64x64 pixels)
  ➢ 4096 \times N

Sparse and Overcomplete Representations
Overcomplete Representations

- What is the dimensionality of the input image? (say 64x64 image)
  - 4096

- What is the dimensionality of the dictionary? (each image = 64x64 pixels)
  - 4096 x N

VERY LARGE!!!
Overcomplete Representations

• What is the dimensionality of the input image? (say 64x64 image)

If \( N > 4096 \) (as it likely is)
we have an overcomplete representation

• What is the dimensionality of the dictionary? (each image = 64x64 pixels)

\[ 4096 \times N \text{ \_ VERY LARGE!!! \_} \]
Overcomplete Representations

- What is the dimensionality of the input image? (say 64x64 image)

More generally:

If #(basis vectors) > dimensions of input

we have an **overcomplete** representation

- 4096 x N

VERY LARGE!!!
Quick Linear Algebra Refresher

• Remember, #(Basis Vectors) = #unknowns

\[ D \alpha = X \]

- Basis Vectors (from Dictionary)
- Weights
- Input data

Sparse and Overcomplete Representations
Recap – Conventional : Images

• Conventional characterization: Images
  – Store Pixel values (positions implicit)

• Image: $128 \times \text{pixel.at.}(1,1) + 134 \times \text{pixel.at.}(1,2) + \ldots + 127 \times \text{pixel.at.}(2,1)\ldots$

• Store only the numbers (128, 134, …, 127)

• *Bases* are “pixel.at.(1,1)”, “pixel.at.(1,2)” etc..
  – Or rather $[1 \ 0 \ 0 \ 0 \ldots]$, $[0 \ 1 \ 0 \ 0 \ldots]$
  – Pixel positions are implicit
Recap – Conventional: Images

- Storing an Image: \[ B.P = \text{Image} \]

Pixel position (rows)

No. of bases = dimension of bases

Basis matrix is square

\[
\begin{pmatrix}
1 & 0 & \cdots \\
0 & 1 & \cdots \\
\vdots & \vdots & \ddots
\end{pmatrix} P = \text{Image}
\]

- “Bases” are unit-valued pixels at specific locations
  - Only weights are stored
  - Basis matrix is \textit{implicit} (everyone knows what it is)
Recap – Conventional: Sound

- Signal = 3 \times \text{sample.at.t=1} + 4 \times \text{sample.at.t=2} + 3.5 \times \text{sample.at.t=3} \ldots
- Store only the numbers [3, 4, 3.5\ldots]
- Bases are “sample.at.t=1”, “sample.at.t=2”, ...
  - Or rather [..0 1 0 0 0 ...], [..0 0 1 0 0 0 ...], ....
  - “Time” is implicit
Recap – Conventional: Sound

• Storing a sound

B.S = Recording

Sample position (rows)

No. of bases = dimension of bases

Basis matrix is square

No. of bases = dimension

“Bases” are unit-valued samples at specific time instants

- Only weights are stored
- Basis matrix is implicit (everyone knows what it is)
Recap: \textit{Component-based} representations

\[
\begin{bmatrix}
  b_{11} & b_{12} & \ldots \\
  b_{21} & b_{22} & \ldots \\
\end{bmatrix} S = \text{Signal}
\]

- Bases may be deterministic, e.g. sinusoids/wavelets or derived, e.g. PCA / ICA / NMF bases
- Only store $w$ to represent individual signals. Bases matrix $B$ stored separately as a one-time deal

Typically, no. of bases (columns) is no more than no. of dimensions

Basis matrix is tall or square
Dictionary based Representations

• Overcomplete “dictionary”-based representations are composition-based representations with more bases than the dimensionality of the data

\[ D \alpha = X \]

Bases matrix is \textit{wide} (more bases than dimensions)
Why Dictionary-based Representations?

• Dictionary based representations are semantically more meaningful

• Enable content-based description
  – Bases can capture entire structures in data
  – E.g. notes in music
  – E.g. image structures (such as faces) in images

• Enable content-based processing
  – Reconstructing, separating, denoising, manipulating speech/music signals
  – Coding, compression, etc.

• Statistical reasons: We will get to that shortly..
Problems

• How to obtain the dictionary
  – Which will give us meaningful representations
• How to compute the weights?
Problems

• How to obtain the dictionary
  – Which will give us meaningful representations

• How to compute the weights?
Quick Linear Algebra Refresher

- Remember, \( \#(\text{Basis Vectors}) = \#\text{unknowns} \)

\[
D\alpha = X
\]

Basis Vectors → Input data

Unknowns
Quick Linear Algebra Refresher

• Remember, #(Basis Vectors) = #unknowns

D.α = X

Basis Vectors

Input data

Unknowns

When can we solve for α?
Quick Linear Algebra Refresher

D.\alpha = X

• When #(Basis Vectors) = dim(Input Data), we have a unique solution
• When #(Basis Vectors) < dim(Input Data), we may have no exact solution
• When #(Basis Vectors) > dim(Input Data), we have infinitely many solutions
Quick Linear Algebra Refresher

\[ D.\alpha = X \]

- When \(#(\text{Basis Vectors}) = \text{dim(Input Data)}\), we have a unique solution.
- When \(#(\text{Basis Vectors}) < \text{dim(Input Data)}\), we may have no solution.
- When \(#(\text{Basis Vectors}) > \text{dim(Input Data)}\), we have infinitely many solutions.

Our Case
Overcomplete Representation

\[ D \alpha = X \]

\#(Basis Vectors) > dimensions of the input
Overcompleteness and Sparsity

• To solve an overcomplete system of the type:

\[ D.\alpha = X \]

• Make assumptions about the data.

• Suppose, we say that \( X \) is composed of no more than a fixed number \( (k) \) of “bases” from \( D \) \( (k \leq \dim(X)) \)
  – The term “bases” is an abuse of terminology.

• Now, we can find the set of \( k \) bases that best fit the data point, \( X \).
Representing Data

Using bases that we know...

But no more than $k=4$ bases
Overcompleteness and Sparsity

But no more than $k=4$ bases are "active"
Overcompleteness and Sparsity

- But no more than $k=4$ bases
No more than 4 bases
No more than 4 bases

ONLY THE $\alpha$ COMPONENTS CORRESPONDING TO THE 4 KEY DICTIONARY ENTRIES ARE NON-ZERO
No more than 4 bases

Only the $\alpha$ components corresponding to the 4 key dictionary entries are non-zero.

Most of $\alpha$ is zero!!

$\alpha$ is sparse.
Sparse representations are representations that account for most or all information of a signal with a linear combination of a small number of atoms.

(from: www.see.ed.ac.uk/~tblumens/Sparse/Sparse.html)
The Sparsity Problem

• We don’t really know $k$
• You are given a signal $X$
• Assuming $X$ was generated using the dictionary, can we find $\alpha$ that generated it?
The Sparsity Problem

- We want to use as few basis vectors as possible to do this.

\[
\begin{align*}
\text{Min} & \quad \|\alpha\|_0 \\
\text{s.t.} & \quad X = \mathbf{D}\alpha
\end{align*}
\]
The Sparsity Problem

- We want to use as few basis vectors as possible to do this.

\[
\begin{align*}
\text{Min} & \quad \|\alpha\|_0 \\
\text{s.t.} & \quad X = D\alpha
\end{align*}
\]

Counts the number of non-zero elements in \(\alpha\)
The Sparsity Problem

• We want to use **as few basis vectors** as possible to do this
  – Ockham’s razor: Choose the simplest explanation invoking the fewest variables

\[
\begin{align*}
\min_{\alpha} & \quad \|\alpha\|_0 \\
\text{s.t.} & \quad X = D\alpha
\end{align*}
\]
The Sparsity Problem

• We want to use as few basis vectors as possible to do this.

\[
\min_{\alpha} \left\| \alpha \right\|_0 \\
\text{s.t. } X = D\alpha
\]

How can we solve the above?
Obtaining Sparse Solutions

• We will look at 2 algorithms:
  – Matching Pursuit (MP)
  – Basis Pursuit (BP)
Matching Pursuit (MP)

- Greedy algorithm
- Finds an atom in the dictionary that best matches the input signal
- Remove the weighted value of this atom from the signal
- Again, find an atom in the dictionary that best matches the remaining signal.
- Continue till a defined stop condition is satisfied.
Matching Pursuit

• Find the dictionary atom that best matches the given signal.

Weight = \( w_1 \)
Matching Pursuit

- Remove weighted image to obtain updated signal

Find best match for this signal from the dictionary
Matching Pursuit

- Find best match for updated signal
Matching Pursuit

- Find best match for updated signal

Iterate till you reach a stopping condition,
\[ \text{norm(ResidualInputSignal)} < \text{threshold} \]

Sparse and Overcomplete Representations
Matching Pursuit

Algorithm Matching Pursuit
Input: Signal: \( f(t) \).
Output: List of coefficients: \( (a_n, g_{\gamma_n}) \).
Initialization:
\[
Rf_1 \leftarrow f(t);
\]
Repeat
\[
\text{find } g_{\gamma_n} \in D \text{ with maximum inner product } < Rf_n, g_{\gamma_n} >;
\]
\[
a_n \leftarrow < Rf_n, g_{\gamma_n} >;
\]
\[
Rf_{n+1} \leftarrow Rf_n - a_n g_{\gamma_n};
\]
\[
n \leftarrow n + 1;
\]
Until stop condition (for example: \( \| Rf_n \| < \text{threshold} \))

From http://en.wikipedia.org/wiki/Matching_pursuit
Matching Pursuit

• Problems ???
Matching Pursuit

• Main Problem
  – Computational complexity
  – The entire dictionary has to be searched at every iteration
# Comparing MP and BP

<table>
<thead>
<tr>
<th>Matching Pursuit</th>
<th>Basis Pursuit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hard thresholding</td>
<td></td>
</tr>
<tr>
<td>(remember the equations)</td>
<td></td>
</tr>
<tr>
<td>Greedy optimization at each step</td>
<td></td>
</tr>
<tr>
<td>Weights obtained using greedy rules</td>
<td></td>
</tr>
</tbody>
</table>

**Sparse and Overcomplete Representations**
Basis Pursuit (BP)

• Remember,

\[
\begin{align*}
\min_{\alpha} & \quad \|\alpha\|_0 \\
\text{s.t.} & \quad X = D\alpha
\end{align*}
\]
• Remember,

\[
\min_{\alpha} \|\alpha\|_0 \\
\text{s.t. } X = D\alpha
\]

In the general case, this is intractable
Basis Pursuit

• Remember,

\[
\begin{align*}
\text{Min} & \quad \|\alpha\|_0 \\
\text{s.t.} & \quad X = D\alpha
\end{align*}
\]

In the general case, this is intractable

Requires combinatorial optimization
Basis Pursuit

- Replace the intractable expression by an expression that is solvable

\[
\min_{\alpha} \|\alpha\|_1 \\
\text{s.t. } X = D\alpha
\]
Basis Pursuit

• Replace the intractable expression by an expression that is solvable

\[
\min_{\alpha} \left\| \alpha \right\|_1 \\
\text{s.t. } X = D\alpha
\]

This will provide identical solutions when \(D\) obeys the Restricted Isometry Property.
Basis Pursuit

• Replace the intractable expression by an expression that is solvable

\[
\min_{\alpha} \|\alpha\|_1 \\
\text{s.t. } X = D\alpha
\]
Basis Pursuit

• We can formulate the optimization term as:

\[
\min_{\alpha} \left\{ \|X - D\alpha\|^2 + \lambda \|\alpha\|_1 \right\}
\]
Basis Pursuit

• We can formulate the optimization term as:

$$\min_{\alpha} \left\{ \| X - D\alpha \|^2 + \lambda \| \alpha \|_1 \right\}$$

$\lambda$ is a penalty term on the non-zero elements and promotes sparsity.
Basis Pursuit

Equivalent to LASSO; for more details, see this paper by Tibshirani

http://www-stat.stanford.edu/~tibs/ftp/lasso.ps

\[
\min_{\alpha} \left\{ \| X - D\alpha \|^2 + \lambda \| \alpha \|_1 \right\}
\]

\( \lambda \) is a penalty term on the non-zero elements and promotes sparsity
Basis Pursuit

\[
\min_{\alpha} \left\{ \| X - D\alpha \|^2 + \lambda \| \alpha \|_1 \right\}
\]

\[
\frac{\partial \| \alpha \|_1}{\partial \alpha_j} = \begin{cases} 
+1 & \text{at } \alpha_j > 0 \\
[-1,1] & \text{at } \alpha_j = 0 \\
-1 & \text{at } \alpha_j < 0 
\end{cases}
\]

• \( \| \alpha \|_1 \) is not differentiable at \( \alpha_j = 0 \)
• Gradient of \( \| \alpha \|_1 \) for gradient descent update
• At optimum, following conditions hold

\[
\nabla_j \| X - D\alpha \|^2 + \lambda \text{sign}(\alpha_j) = 0, \quad \text{if } |\alpha_j| > 0
\]

\[
\nabla_j \| X - D\alpha \|^2 \leq \lambda, \quad \text{if } \alpha_j = 0
\]
Basis Pursuit

• There are efficient ways to solve the LASSO formulation.  
  – http://web.stanford.edu/~hastie/glmnet_matlab/

• Simplest solution: Coordinate descent algorithms  
  – On webpage..
L₁ vs L₀

Minimize $\|\alpha\|_0$

subject to $X = D\alpha$

• L₀ minimization
  – Two-sparse solution
  – ANY pair of bases can explain $X$ with 0 error
• **$L_1$ minimization**
  
  – Two-sparse solution
  
  – All else being equal, the two closest bases are chosen
Comparing MP and BP

<table>
<thead>
<tr>
<th>Matching Pursuit</th>
<th>Basis Pursuit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hard thresholding</td>
<td>Soft thresholding</td>
</tr>
<tr>
<td>(remember the equations)</td>
<td></td>
</tr>
<tr>
<td>Greedy optimization at each step</td>
<td>Global optimization</td>
</tr>
<tr>
<td>Weights obtained using greedy rules</td>
<td>Can force N-sparsity with appropriately chosen weights</td>
</tr>
</tbody>
</table>
General Formalisms

- $L_0$ minimization
  \[ \min_{\alpha} \|\alpha\|_0 \]
  \[ \text{s.t. } X = D\alpha \]

- $L_0$ constrained optimization
  \[ \min_{\alpha} \|X - D\alpha\|^2 \]
  \[ \text{s.t. } \|\alpha\|_0 < C \]

- $L_1$ minimization
  \[ \min_{\alpha} \|\alpha\|_1 \]
  \[ \text{s.t. } X = D\alpha \]

- $L_1$ constrained optimization
  \[ \min_{\alpha} \|X - D\alpha\|^2 \]
  \[ \text{s.t. } \|\alpha\|_1 < C \]
Many Other Methods..

• Iterative Hard Thresholding (IHT)
• CoSAMP
• OMP
• ...

Sparse and Overcomplete Representations
Problems

• How to obtain the dictionary
  – Which will give us meaningful representations

• How to compute the weights?
Dictionaries: Compressive Sensing

• Just random vectors!
More Structured ways of Constructing Dictionaries

• Dictionary entries must be structurally “meaningful”
  – Represent true compositional units of data

• Have already encountered two ways of building dictionaries
  – NMF for non-negative data
  – K-means ..
K-Means for Composing Dictionaries

• Every vector is approximated by the centroid of the cluster it falls into

• Cluster means are “codebook” entries
  – Dictionary entries
  – Also compositional units the compose the data

Train the codebook from training data using K-means
K-Means for Dictionaries

- $\alpha$ must be 1 sparse
- Only $\alpha$ entry must 1

$\|\alpha\|_0 = 1 \quad \|\alpha\|_1 = 1$

Each column is a codeword (centroid) from the codebook

Sparse and Overcomplete Representations
• Learn Codewords to minimize the total squared length of the training vectors from the closest codeword.
**Length-unconstrained**

**K-Means for Dictionaries**

- $\alpha$ must be 1 sparse
- No restriction on $\alpha$ value

$\|\alpha\|_0 = 1$

Each column is a codeword (centroid) from the codebook.
SVD K-Means

• Learn Codewords to minimize the total squared projection error of the training vectors from the closest codeword
SVD K-means

1. Initialize a set of centroids randomly

2. For each data point $x$, find the projection from the centroid for each cluster
   - $p_{\text{cluster}} = |x^T m_{\text{cluster}}|

3. Put data point in the cluster of the closest centroid
   - Cluster for which $p_{\text{cluster}}$ is maximum

4. When all data points are clustered, recompute centroids

$$m_{\text{cluster}} = \text{Principal Eigenvector}\left(\{x \mid x \in \text{cluster}\}\right)$$
Problem

• Only represents *Radial* patterns
What about this pattern?

- Dictionary entries that represent radial patterns will not capture this structure
  - 1-sparse representations will not do
What about this pattern?

• We need **AFFINE** patterns
What about this pattern?

- We need **AFFINE** patterns
- *Each vector is modeled by a linear combination of K (here 2) bases*
What about this pattern?

• We need **AFFINE patterns**

• *Each vector is modeled by a linear combination of K (here 2) bases*

Every line is a (constrained) combination of two bases

**2-sparse**

Constraint:
Line = a.b₁ + (1-a)b₂
Each column is a codeword (centroid) from the codebook

- \( \alpha \) must be \( k \) sparse
- No restriction on \( \alpha \) value

\[ \| \alpha \|_0 = k \]
K SVD

• Initialize Codebook

1. For every vector, compute K-sparse alphas
   – Using any pursuit algorithm

\[
\begin{align*}
D &= \begin{bmatrix}
0 & 0 & 2 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 \\
0 & 7 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 \\
3 & 0 & 0 & 0 & 0 \\
0 & 5 & 0 & 0 & 1 \\
0 & 0 & 1 & 0 & 2
\end{bmatrix}
\end{align*}
\]

\[
\alpha &= \begin{bmatrix}
0 & 0 & 2 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 \\
0 & 7 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 \\
3 & 0 & 0 & 0 & 0 \\
0 & 5 & 0 & 0 & 1 \\
0 & 0 & 1 & 0 & 2
\end{bmatrix}
\]
**K-SVD**

2. For each codeword \((k)\):
   - For each vector \(x\)
     - Subtract the contribution of all other codewords to obtain \(e_k(x)\)
     - Codeword-specific residual
   - Compute the principal Eigen vector of \(\{e_k(x)\}\)

3. Return to step 1

\[
e_k(x) = x - \sum_{j \neq k} \alpha_j D_j
\]
K-SVD

• Termination of each iteration: Updated dictionary

• Conclusion: A dictionary where any data vector can be composed of at most K dictionary entries
  – More generally, sparse composition
Problems

• How to obtain the dictionary
  – Which will give us meaningful representations
• How to compute the weights?
Applications of Sparse Representations

• Many many applications
  – Signal representation
  – Statistical modelling
  – ..
  – We’ve seen one: Compressive sensing

• Another popular use
  – Denoising
Denoising

• As the name suggests, remove noise!
Denoising

• As the name suggests, remove noise!
• We will look at image denoising as an example
Image Denoising

• Here’s what we want
Image Denoising

• Here’s what we want
Image Denoising

• Here’s what we want
The Image Denoising Problem

• Given an image
• Remove Gaussian additive noise from it
Image Denoising

\[ Y = X + \varepsilon \]

- **Noisy Input**
- **Orig. Image**
- **Gaussian Noise**

\[ \varepsilon = N(0, \sigma) \]
Image Denoising

• Remove the noise from $\mathbf{Y}$, to obtain $\mathbf{X}$ as best as possible.
Image Denoising

• Remove the noise from $Y$, to obtain $X$ as best as possible

• Using sparse representations over learned dictionaries
Image Denoising

• Remove the noise from $Y$, to obtain $X$ as best as possible
• Using sparse representations over learned dictionaries
• We will *learn* the dictionaries
Image Denoising

• Remove the noise from $Y$, to obtain $X$ as best as possible
• Using sparse representations over learned dictionaries
• We will *learn* the dictionaries
• What data will we use? *The corrupted image itself!*
Image Denoising

• We use the data to be denoised to learn the dictionary.
• Training and denoising become an iterated process.
• We use image patches of size $\sqrt{n} \times \sqrt{n}$ pixels (i.e. if the image is 64x64, patches are 8x8)
Image Denoising

- The data dictionary $D$
  - Size = $n \times k$ ($k > n$)
  - This is known and fixed, to start with
  - Every image patch can be sparsely represented using $D$
Image Denoising

• Recall our equations from before.
• We want to find $\alpha$ so as to minimize the value of the equation below:

\[
\min_{\alpha} \left\{ \|X - D\alpha\|^2 + \lambda \|\alpha\|_0 \right\}
\]

\[
\min_{\alpha} \left\{ \|X - D\alpha\|^2 + \lambda \|\alpha\|_1 \right\}
\]
Image Denoising

\[
\min_\alpha \left\{ \|X - D\alpha\|^2 + \lambda \|\alpha\|_1 \right\}
\]

- In the above, \(X\) is a patch.
Image Denoising

\[ \min_{\alpha} \left\{ \|X - D\alpha\|^2 + \lambda \|\alpha\|_1 \right\} \]

• In the above, \(X\) is a patch.
• If the larger image is fully expressed by the every patch in it, how can we go from patches to the image?
Image Denoising

\[ \text{Min}_{\alpha_{ij}, X} \{ \mu \| X - Y \|_2^2 + \sum_{ij} \| R_{ij} X - D \alpha_{ij} \|_2^2 \} + \sum_{ij} \lambda_{ij} \| \alpha_{ij} \|_0 \} \]
Image Denoising

\[
\min_{\alpha_{ij}, x} \left\{ \mu \left\| X - Y \right\|^2_2 + \sum_{ij} \left\| R_{ij} x - D \alpha_{ij} \right\|^2_2 + \sum_{ij} \lambda_{ij} \left\| \alpha_{ij} \right\|_0 \right\}
\]

\( (X - Y) \) is the error between the input and denoised image. \( \mu \) is a penalty on the error.
Image Denoising

\[
\min_{\alpha_{ij}, X} \left\{ \mu \left\| X - Y \right\|^2_2 + \sum_{ij} \left\| R_{ij}X - D\alpha_{ij} \right\|^2_2 \right\} + \sum_{ij} \lambda_{ij} \left\| \alpha_{ij} \right\|_0
\]

Error bounding in each patch
- \( R_{ij} \) selects the \((ij)^{th}\) patch
- Terms in summation = no. of patches
Image Denoising

\[
\min_{\alpha_{ij}, X} \left\{ \mu \left\| X - Y \right\|_2^2 + \sum_{ij} \left\| R_{ij} X - D \alpha_{ij} \right\|_2^2 \right\} + \sum_{ij} \lambda_{ij} \left\| \alpha_{ij} \right\|_0
\]

\( \lambda \) forces sparsity

Sparse and Overcomplete Representations
Image Denoising

• But, we don’t “know” our dictionary D.
• We want to estimate D as well.
Image Denoising

- But, we don’t “know” our dictionary $D$.
- We want to estimate $D$ as well.

$$\min_{D, \alpha_{ij}, X} \left\{ \mu \|X - Y\|^2_2 + \sum_{ij} \|R_{ij}X - D\alpha_{ij}\|^2_2 \right. + \left. \sum_{ij} \lambda_{ij} \|\alpha_{ij}\|_0 \right\}$$

We can use the previous equation itself!!!
Image Denoising

\[
\min_{D, \alpha_{ij}, X} \left\{ \mu \| X - Y \|_2^2 + \sum_{ij} \| R_{ij} X - D \alpha_{ij} \|_2^2 \right. \\
\left. + \sum_{ij} \lambda_{ij} \| \alpha_{ij} \|_0 \right\}
\]

How do we estimate all 3 at once?
Image Denoising

\[
\text{Min}_{D, \alpha_{ij}, X} \left\{ \mu \| X - Y \|_2^2 + \sum_{ij} \| R_{ij} X - D \alpha_{ij} \|_2^2 \right. \\
\left. + \sum_{ij} \lambda_{ij} \| \alpha_{ij} \|_0 \right\}
\]

How do we estimate all 3 at once?

We cannot estimate them at the same time!
Image Denoising

\[
\min_{D, \alpha_{ij}, X} \{ \mu \| X - Y \|^2_2 + \sum_{ij} \| R_{ij} X - D \alpha_{ij} \|^2_2 \\
+ \sum_{ij} \lambda_{ij} \| \alpha_{ij} \|_0 \} \]

How do we estimate all 3 at once?
Fix 2, and find the optimal 3\textsuperscript{rd}.
Image Denoising

\[
\min_{D, \alpha_{ij}, X} \left\{ \mu \|X - Y\|^2 + \sum_{ij} \|R_{ij}X - D\alpha_{ij}\|^2 + \sum_{ij} \lambda_{ij} \|\alpha_{ij}\|_0 \right\}
\]

Initialize \( X = Y \)
Image Denoising

\[
\min_{\alpha_{ij}} \left\{ \mu \|X - Y\|_2^2 + \sum_{ij} \|R_{ij}X - D\alpha_{ij}\|_2^2 \right. \\
+ \sum_{ij} \lambda_{ij} \|\alpha_{ij}\|_0 \right\}
\]

Initialize \( X = Y \), initialize \( D \)

You know how to solve the remaining portion for \( \alpha \) – MP, BP!
Image Denoising

• Now, update the dictionary D.
• Update D one column at a time, following the K-SVD algorithm
• K-SVD maintains the sparsity structure
Now, update the dictionary $D$. 
Update $D$ one column at a time, following the \textbf{K-SVD algorithm}.

K-SVD maintains the sparsity structure.
Iteratively update $\alpha$ and $D$. 
Image Denoising

Learned Dictionary for Face Image denoising

Image Denoising

\[
\min_X \left\{ \mu \|X - Y\|_2^2 + \sum_{ij} \|R_{ij}X - D\alpha_{ij}\|_2^2 \right\} + \sum_{ij} \lambda_{ij} \|\alpha_{ij}\|_0 \right} \rightarrow \text{Const. wrt X}
\]

We know \(D\) and \(\alpha\)

The quadratic term above has a closed-form solution
Image Denoising

\[
\min_X \{ \mu \|X - Y\|_2^2 + \sum_{ij} \|R_{ij}X - D\alpha_{ij}\|_2^2 + \sum_{ij} \lambda_{ij} \|\alpha_{ij}\|_0 \}
\]

\[
X = (\mu I + \sum_{ij} R_{ij}^T R_{ij})^{-1} (\mu Y + \sum_{ij} R_{ij}^T D\alpha_{ij})
\]

We know D and \(\alpha\)

Sparse and Overcomplete Representations
Image Denoising

• Summarizing... We wanted to obtain 3 things
Image Denoising

• Summarizing... We wanted to obtain 3 things
  ➢ Weights $\alpha$
  ➢ Dictionary $\mathbf{D}$
  ➢ Denoised Image $\mathbf{X}$
Image Denoising

• Summarizing... We wanted to obtain 3 things

➤ Weights $\alpha$ – Your favorite pursuit algorithm
➤ Dictionary $D$ – Using K-SVD
➤ Denoised Image $X$
Image Denoising

- Summarizing... We wanted to obtain 3 things
  - Weights $\alpha$ – Your favorite pursuit algorithm
  - Dictionary $\mathbf{D}$ – Using K-SVD
  - Denoised Image $\mathbf{X}$
Image Denoising

• Summarizing... We wanted to obtain 3 things

➢ Weights $\alpha$
➢ Dictionary $D$
➢ Denoised Image $X$- Closed form solution
Image Denoising

• Here’s what we want
Image Denoising

• Here’s what we want
Comparing to Other Techniques

Non-Gaussian data

PCA of ICA
Which is which?

Comparing to Other Techniques

Non-Gaussian data

Comparing to Other Techniques

Non-Gaussian data

- **Predicts data here**
- **Doesn’t predict data here**
- **Does pretty well**

Comparing to Other Techniques

Data still in 2-D space

ICA

Overcomplete

Doesn’t capture the underlying representation, which Overcomplete representations can do...
Summary

• Overcomplete representations can be more powerful than component analysis techniques.
• Dictionary can be learned from data.
• Relative advantages and disadvantages of the pursuit algorithms.