Machine Learning for Signal Processing
Sparse and Overcomplete Representations

Bhiksha Raj
(slides from Sourish Chaudhuri)

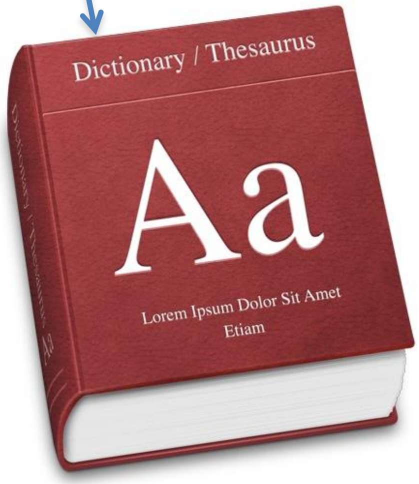
Oct 22, 2015

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?

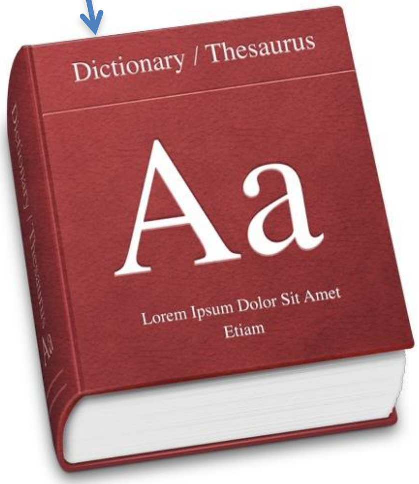
Representing Data

Dictionary (codebook)

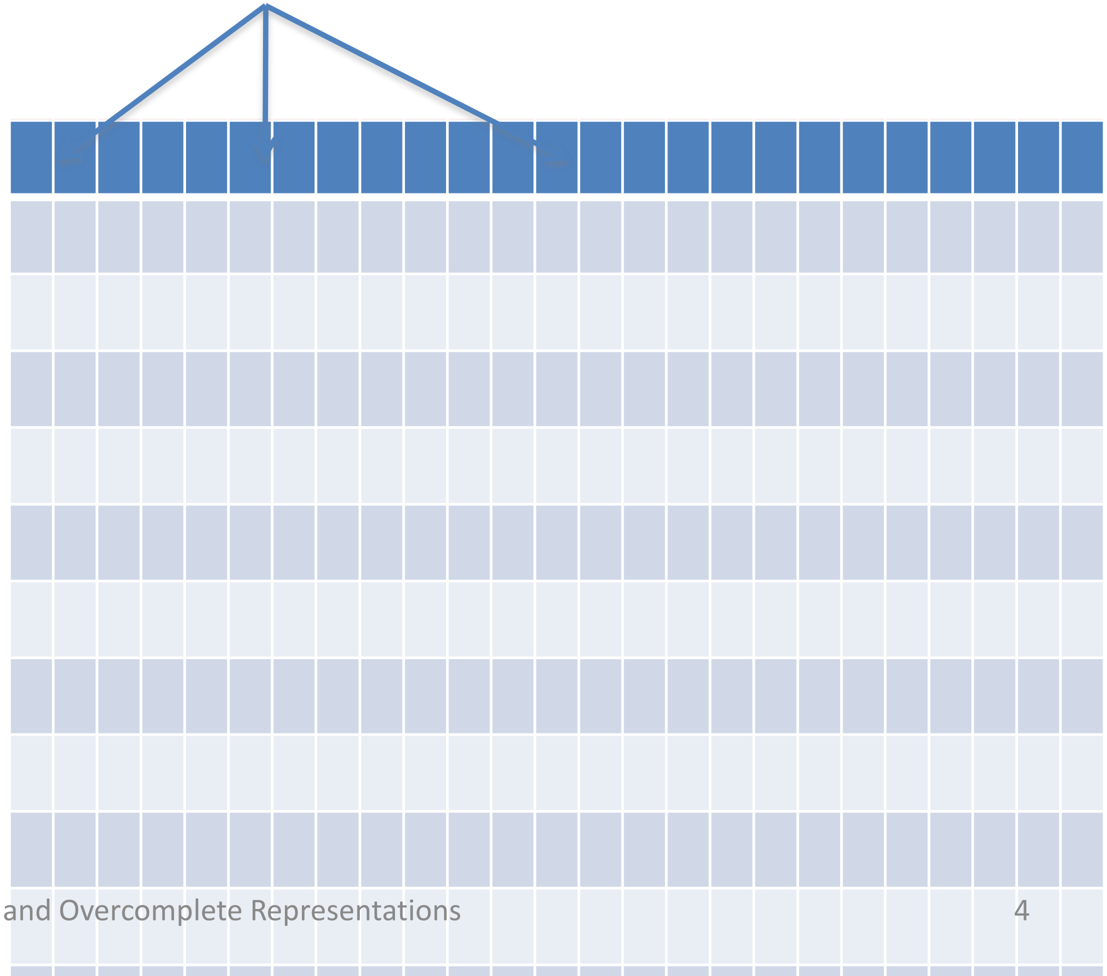


Representing Data

Dictionary

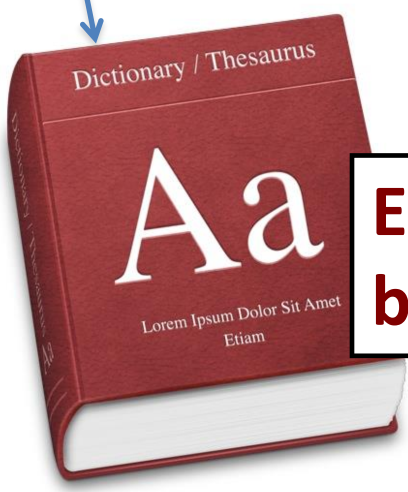


Atoms

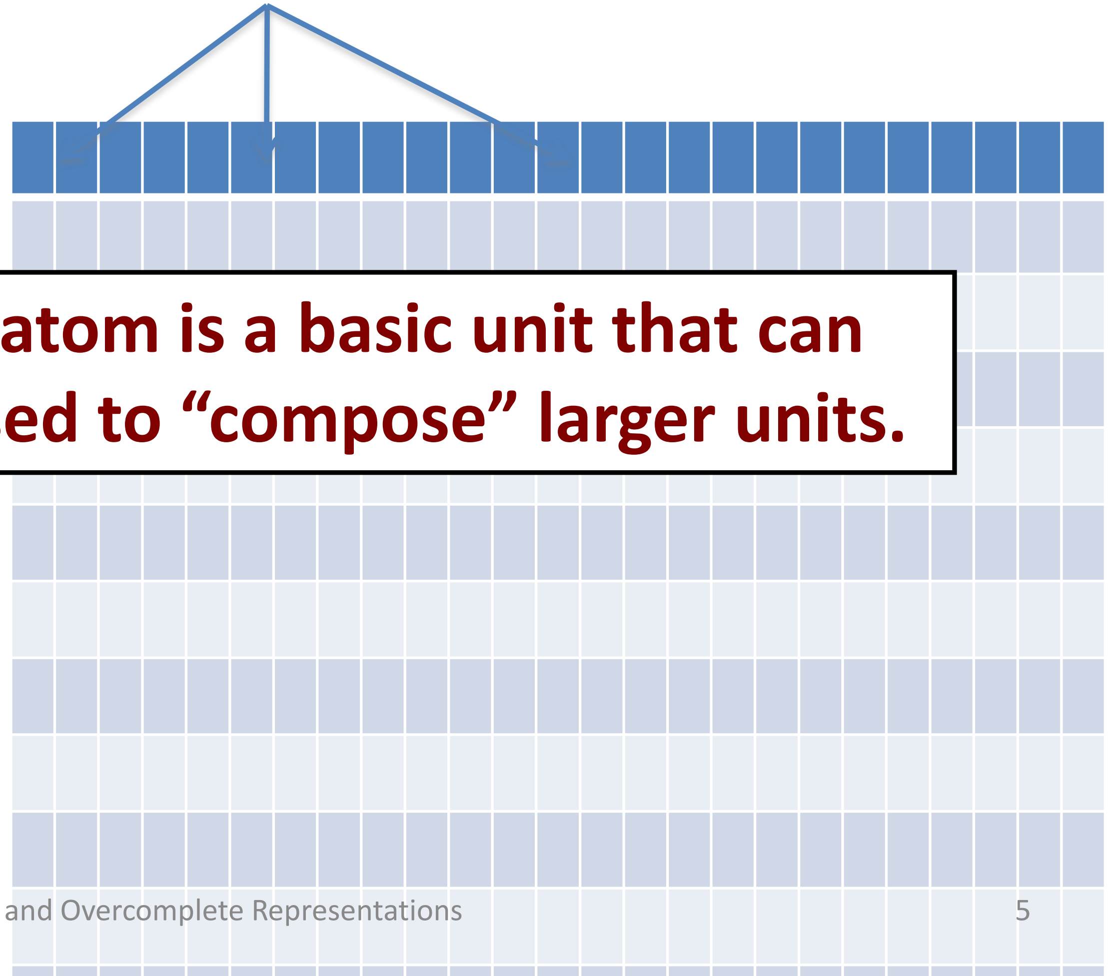


Representing Data

Dictionary

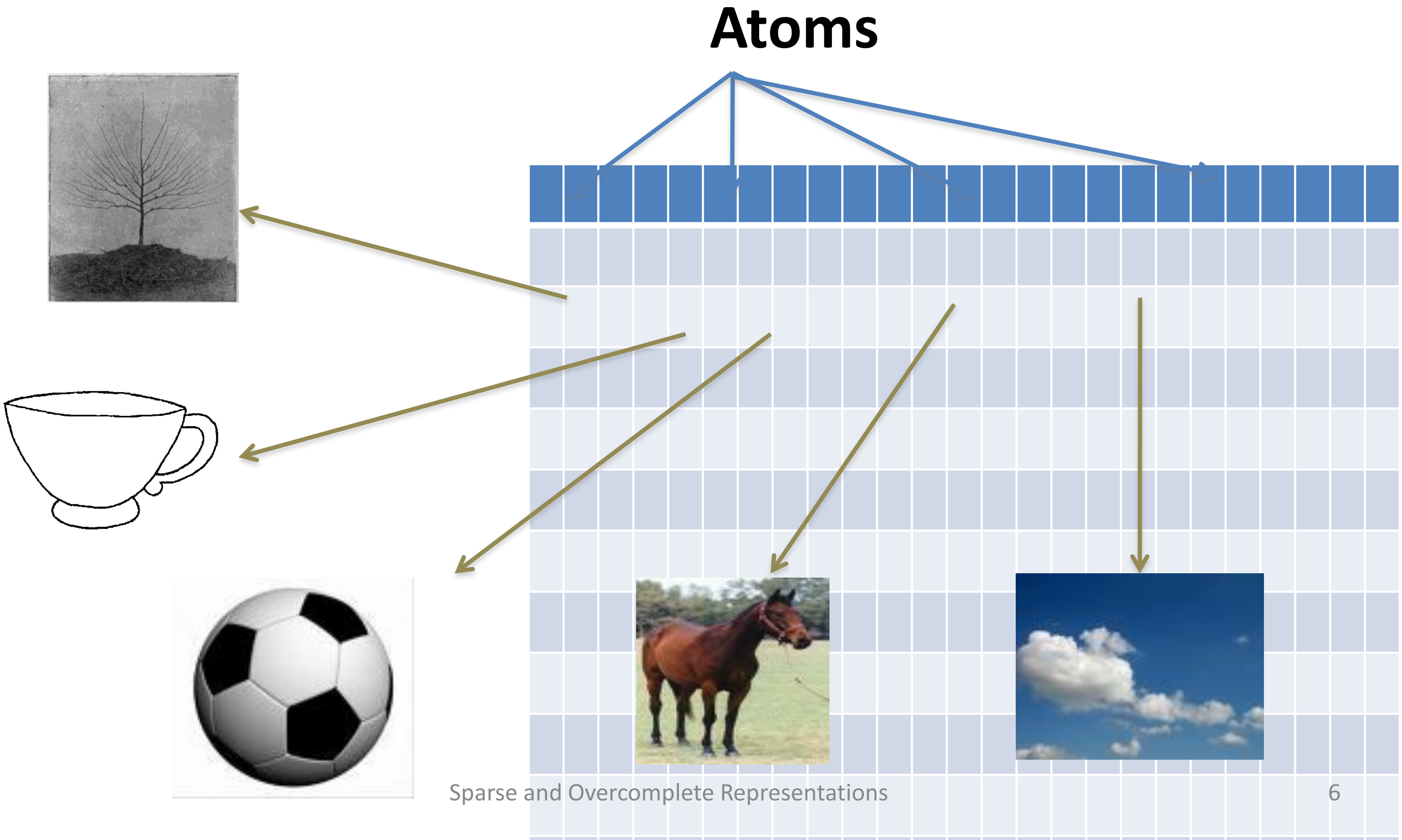


Atoms

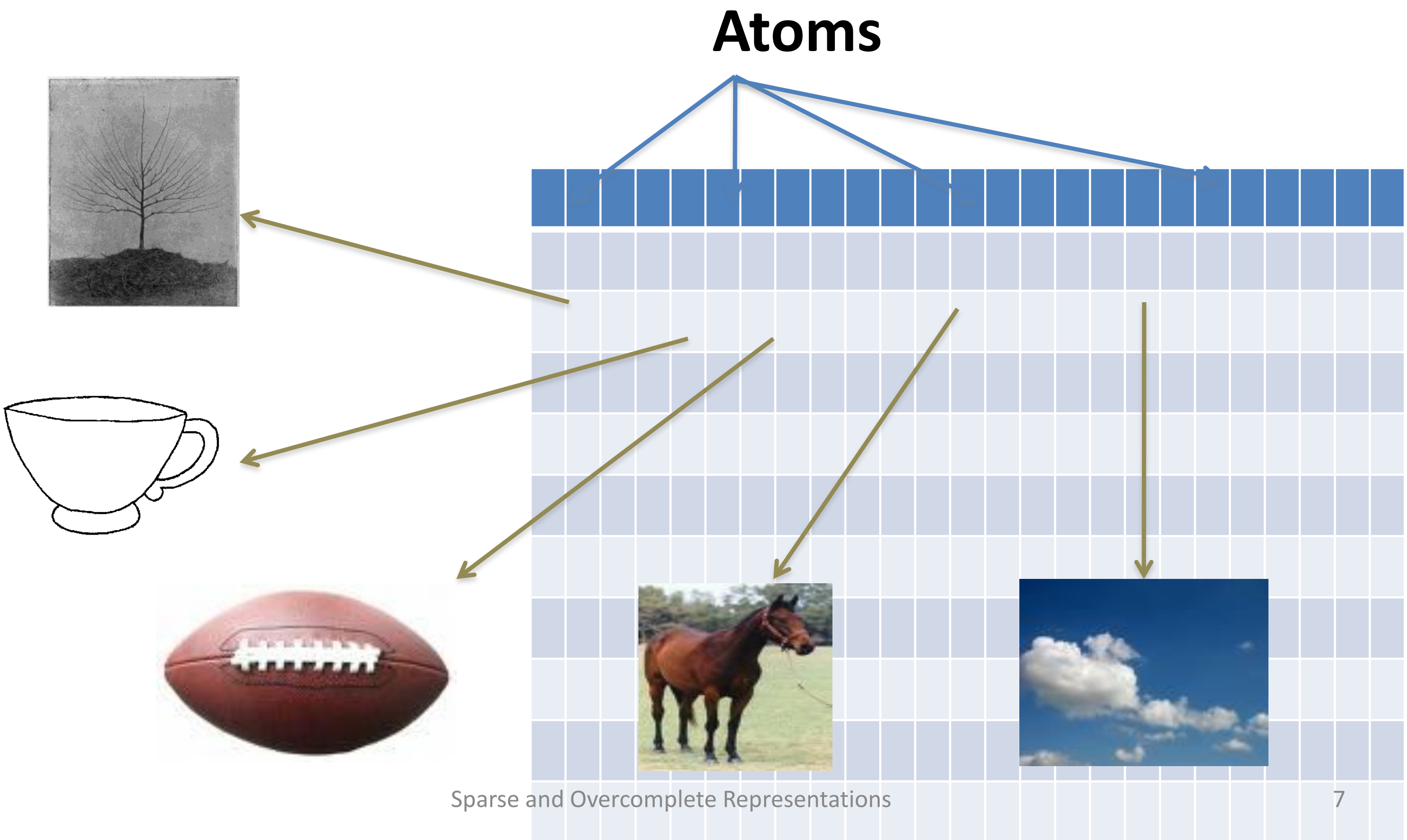


Each atom is a basic unit that can be used to "compose" larger units.

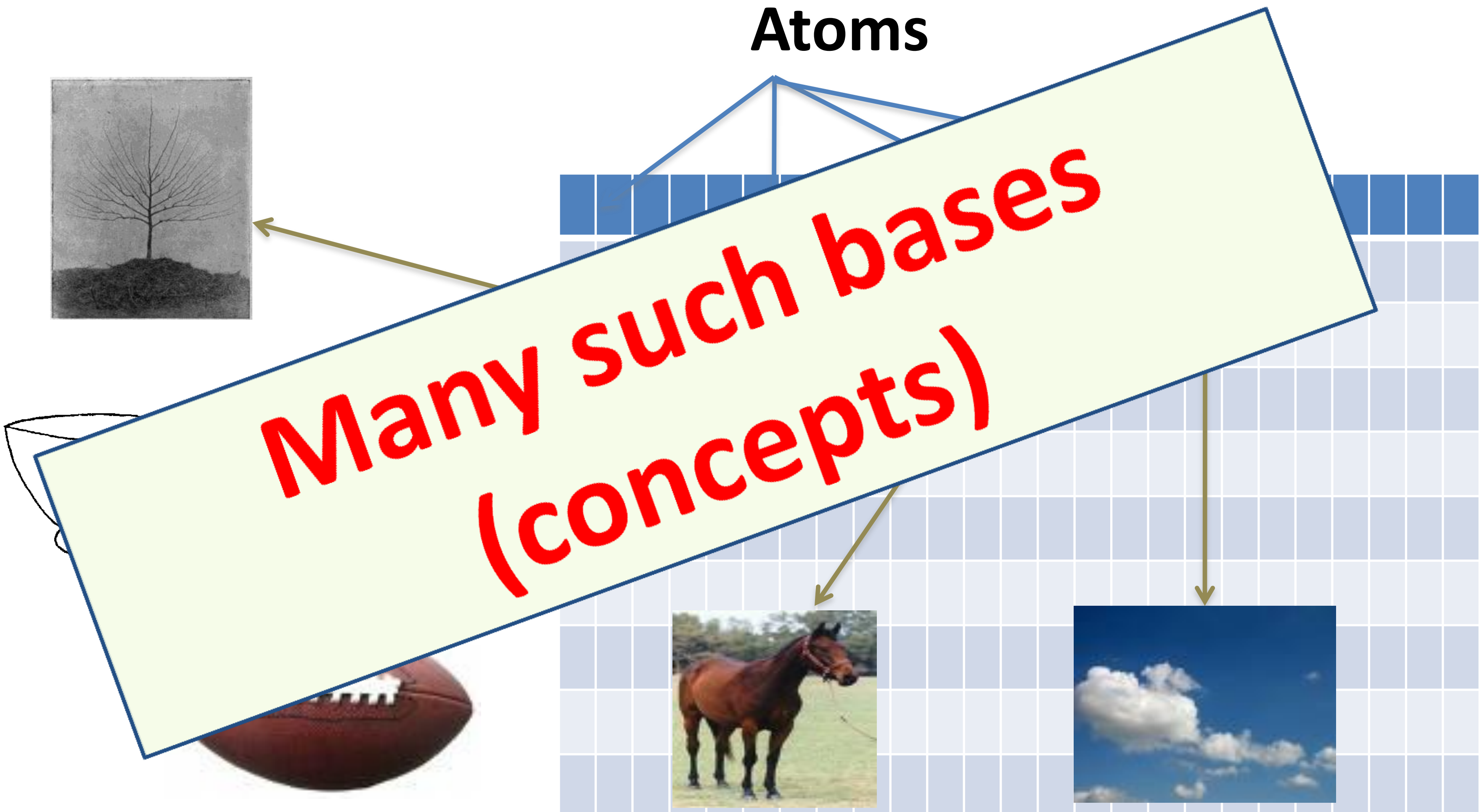
Representing Data



Representing Data



Representing Data



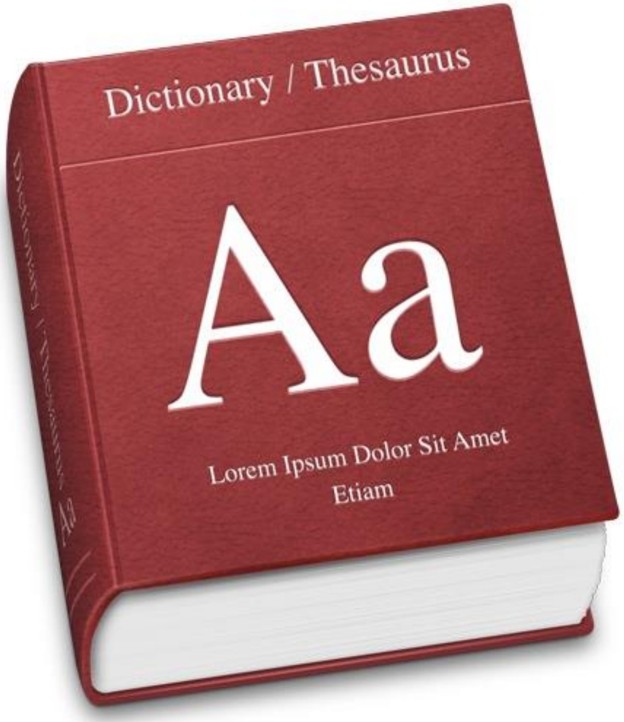
Representing Data

Representing Data

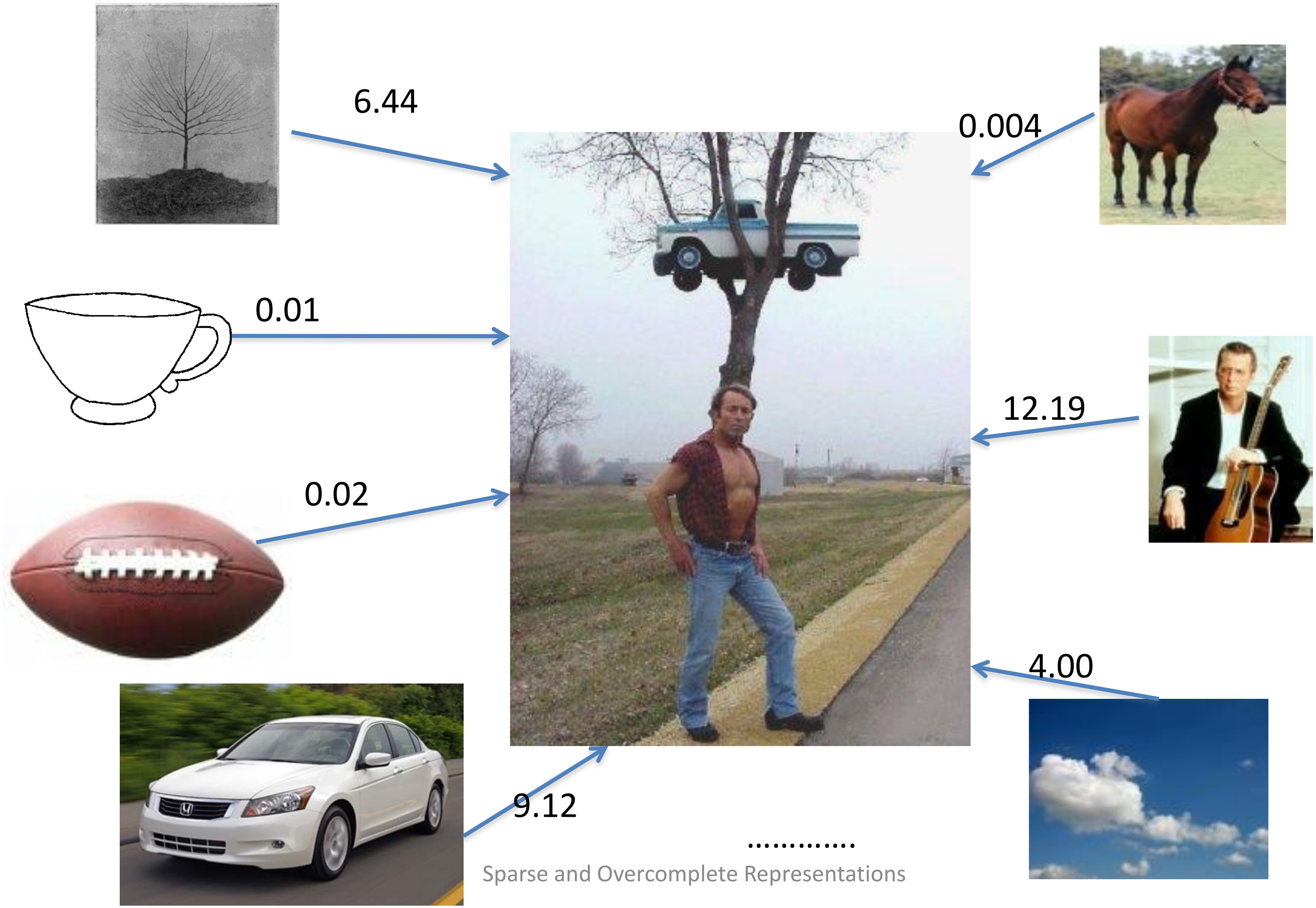
Using concepts that we know…

Representing Data

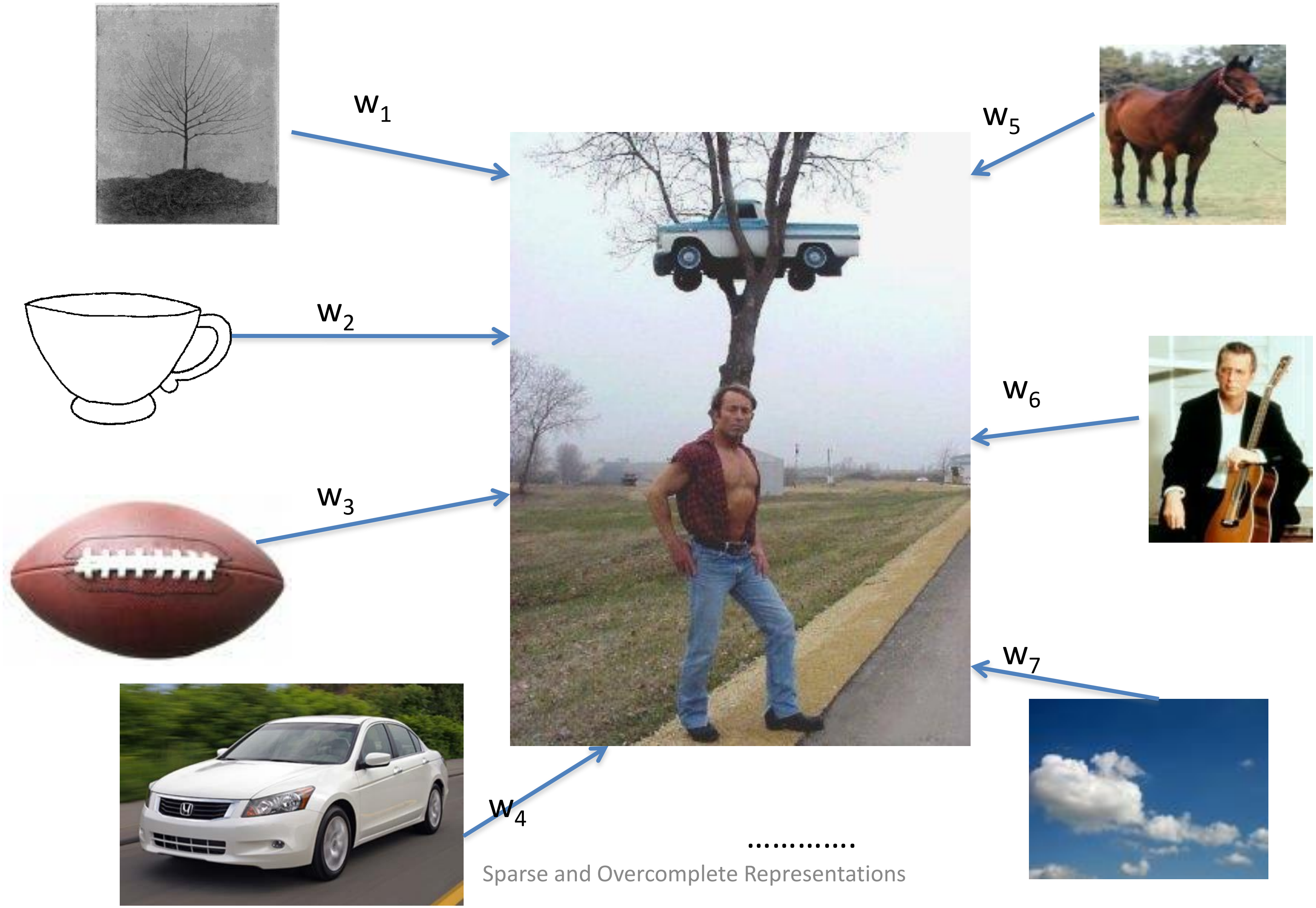
Using concepts that we know…



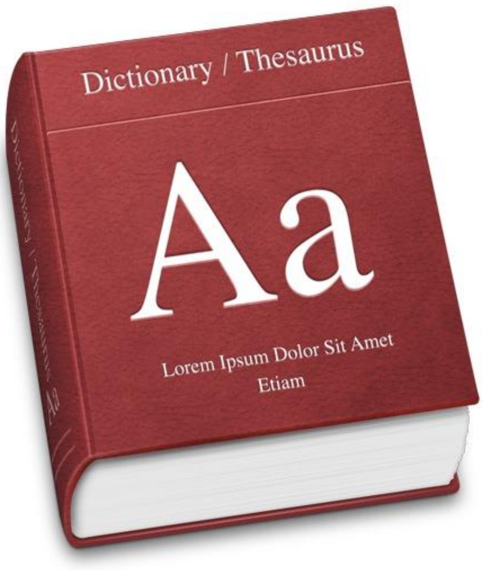
Representing Data



Representing Data

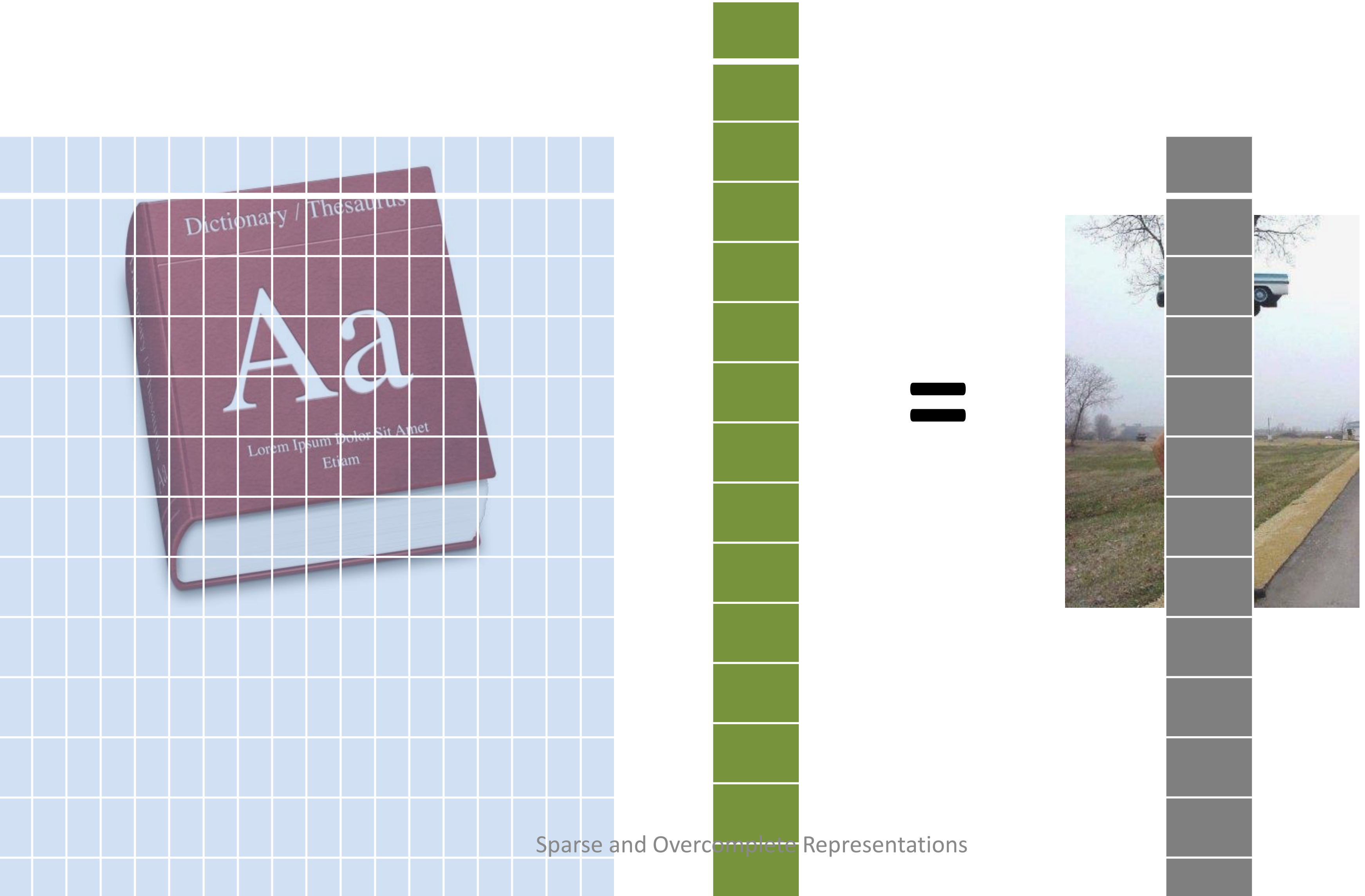


Representing Data

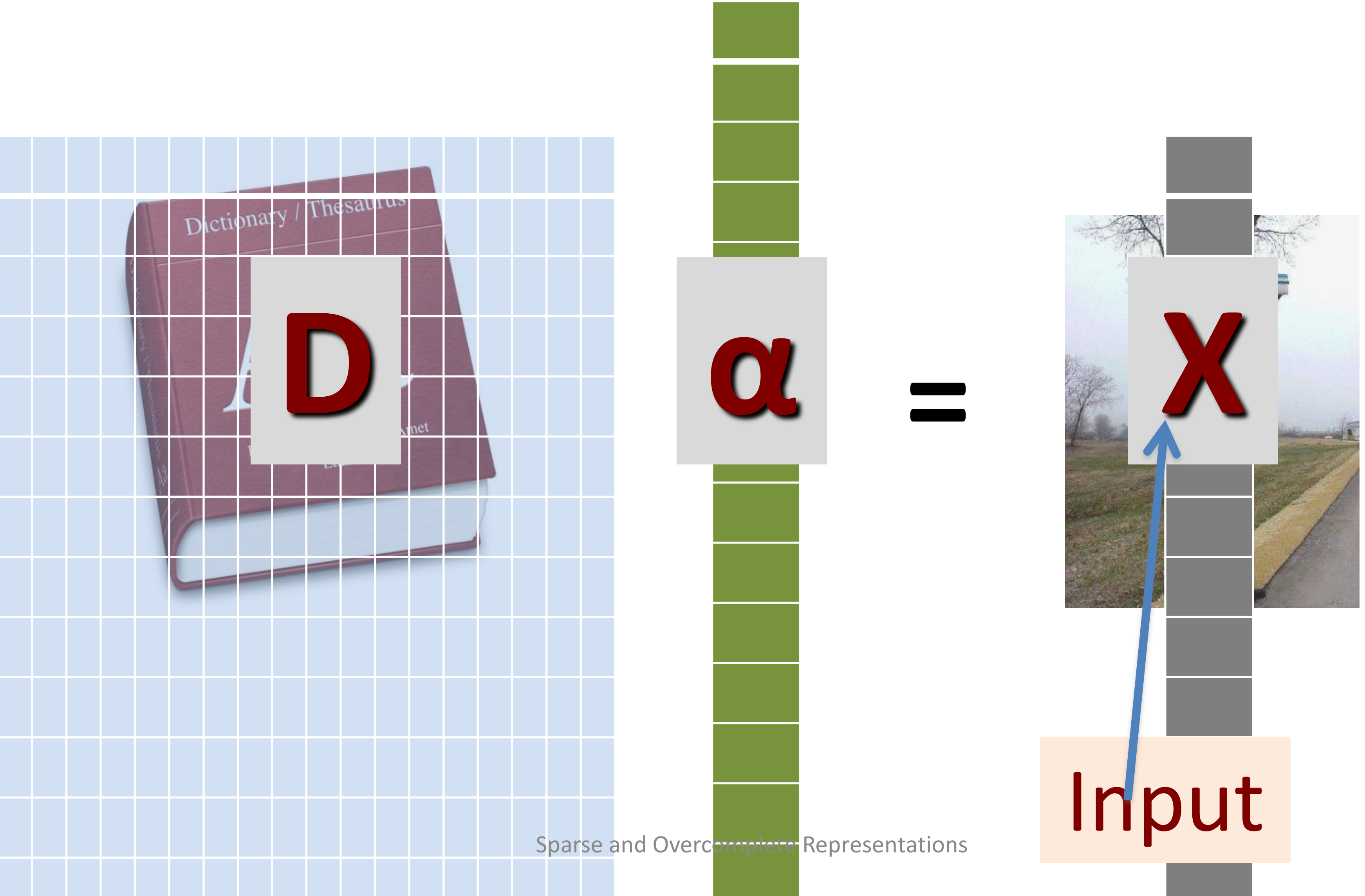


=

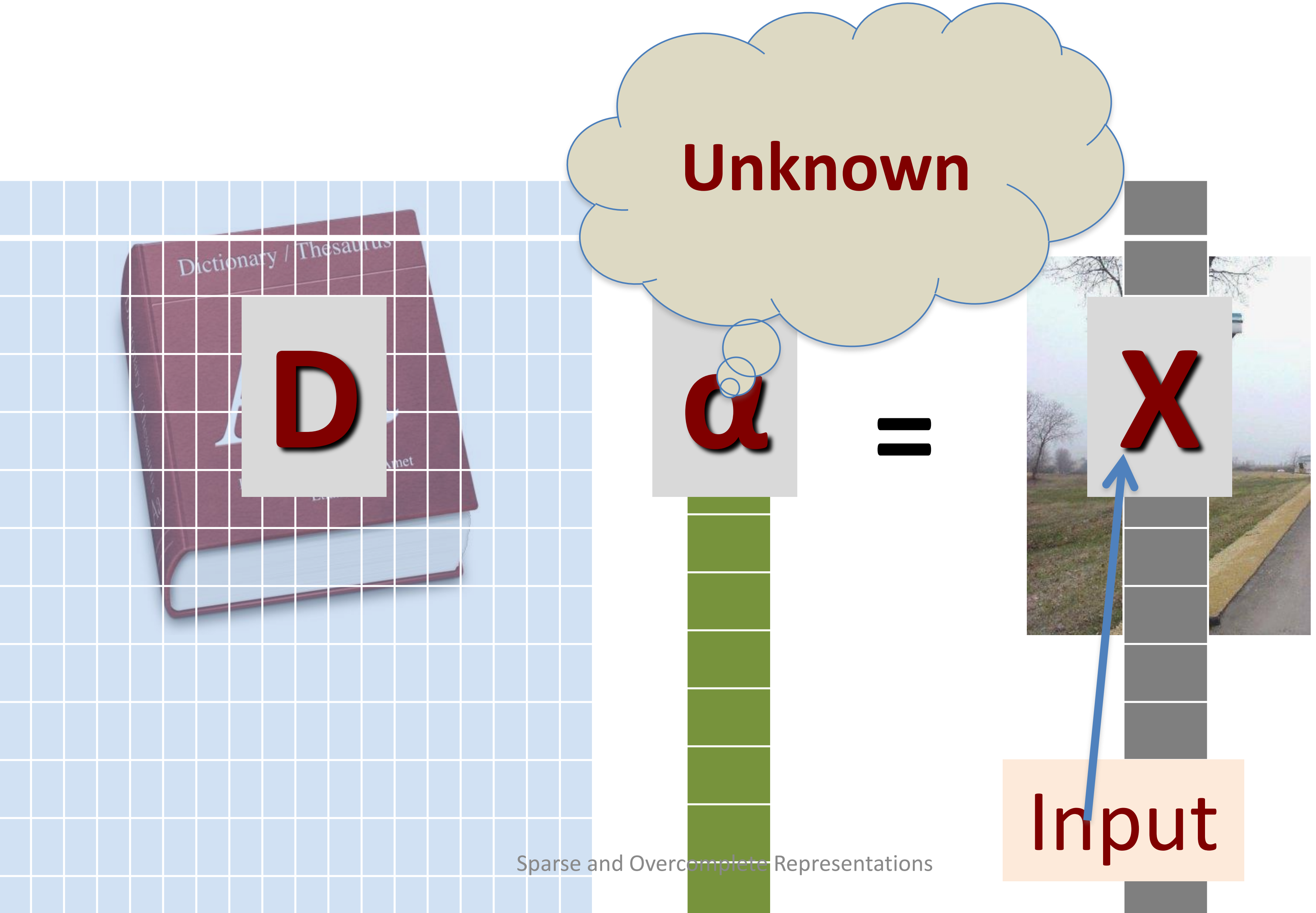
Representing Data



Representing Data



Representing Data



Quick Linear Algebra Refresher

- Remember, #(Basis Vectors)= #unknowns

$$\mathbf{D}.\boldsymbol{\alpha} = \mathbf{X}$$

Basis Vectors (*from Dictionary*)

Weights

Input data

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 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 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 x N**

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

$$D.\alpha = X$$

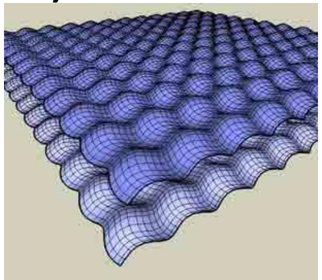
Basis Vectors *(from Dictionary)*

Weights

Input data

Recap – Conventional : Images

- Conventional characterization: Images
  - Store Pixel values (positions implicit)



- Image: 128 x pixel.at.(1,1) + 134 x pixel.at.(1,2) + … + 127 x pixel.at.(2,1)….
- Store only the numbers (128,134, …, 127)
- *Bases* are "pixel.at.(1,1)", "pixel.at.(1,2)" etc..
  - Or rather [1 0 0 0…], [0 1 0 0 …]
  - Pixel positions are implicit

Recap – Conventional : Images

- Storing an Image **B.P = Image**

Pixel position (rows)

$$\begin{pmatrix} 1 & 0 & \cdots \\ 0 & 1 & \cdots \\ \vdots & \vdots & \vdots \end{pmatrix} P = Image$$

No. of bases = dimension of bases

Basis matrix is *square*

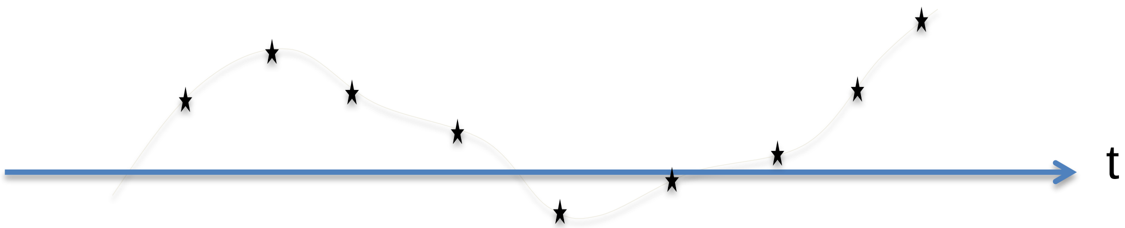
Bases (columns)

Stored Pixel values

Actual image

- "Bases" are unit-valued pixels at specific locations
  - Only weights are stored
  - Basis matrix is *implicit* (everyone knows what it is)

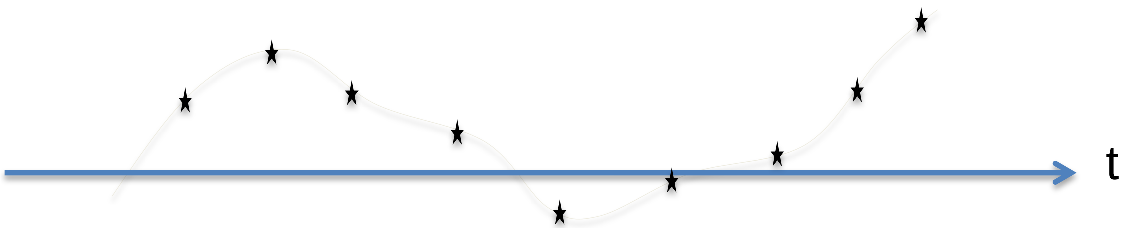
Recap – Conventional: Sound



- Signal = 3 x sample.at.t=1 + 4 x sample.at.t=2 + 3.5 x sample.at.t=3 ....
- Store only the numbers [3, 4, 3.5...]
- 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)

$$\begin{pmatrix} 1 & 0 & \cdots \\ 0 & 1 & \cdots \\ \vdots & \vdots & \vdots \end{pmatrix} S = Signal$$

No. of bases = dimension of bases

Basis matrix is *square*

Bases (columns)

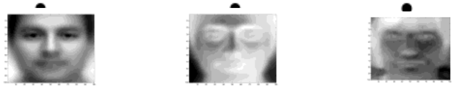
Stored signal values

Actual signal

- "Bases" are unit-valued samples at specific time instants
  - Only weights are stored
  - Basis matrix is *implicit* (everyone knows what it is)

Recap: *Component-based* representations

$$\begin{bmatrix} b_{11} & b_{12} & \cdots \\ b_{21} & b_{22} & \cdots \\ \vdots & \vdots & \vdots \end{bmatrix} S = Signal$$



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

Basis matrix is tall or square

Bases (columns)

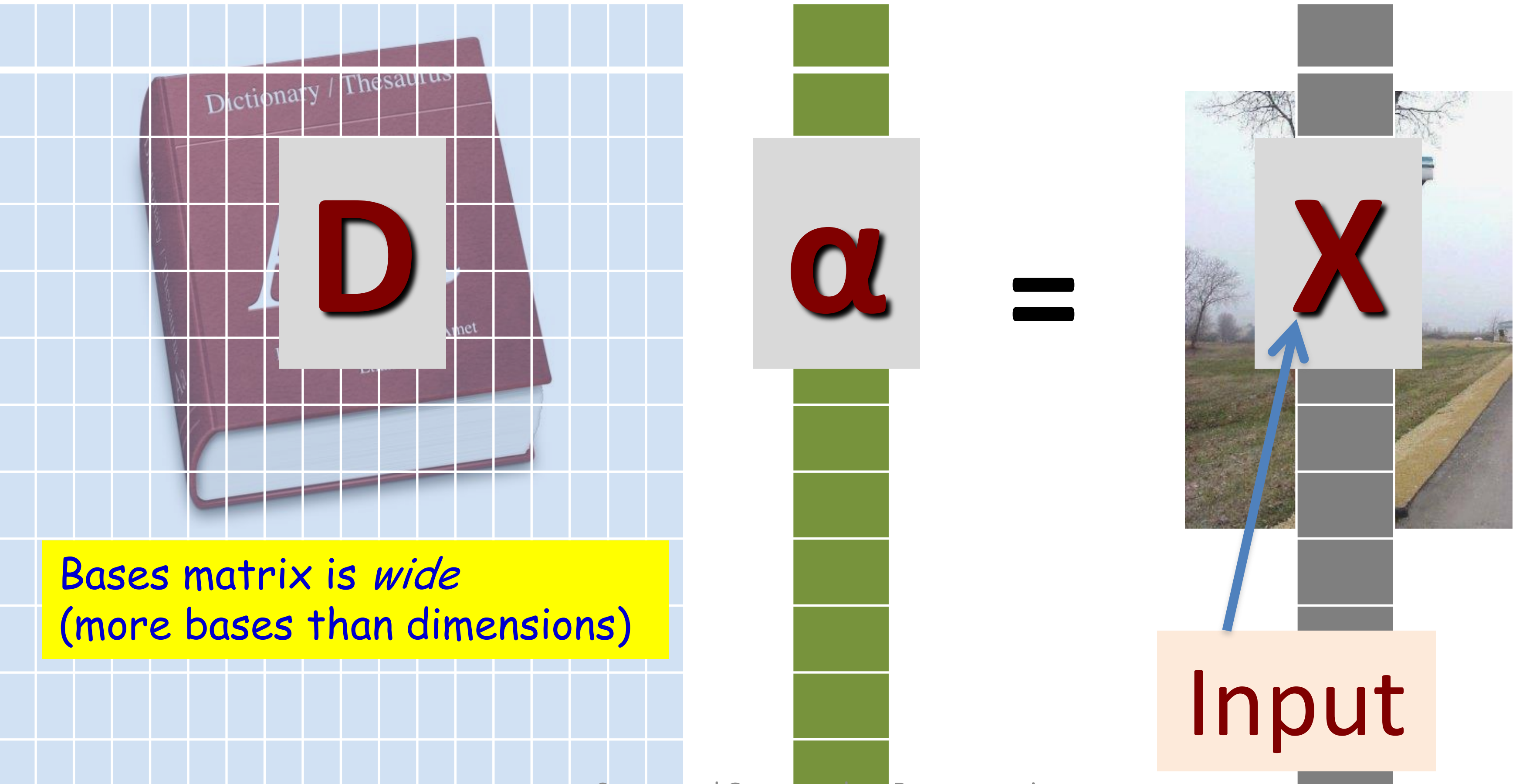
Stored signal values

Actual 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

Dictionary based Representations

- Overcomplete "dictionary"-based representations are composition-based representations with more bases than the dimensionality of the data

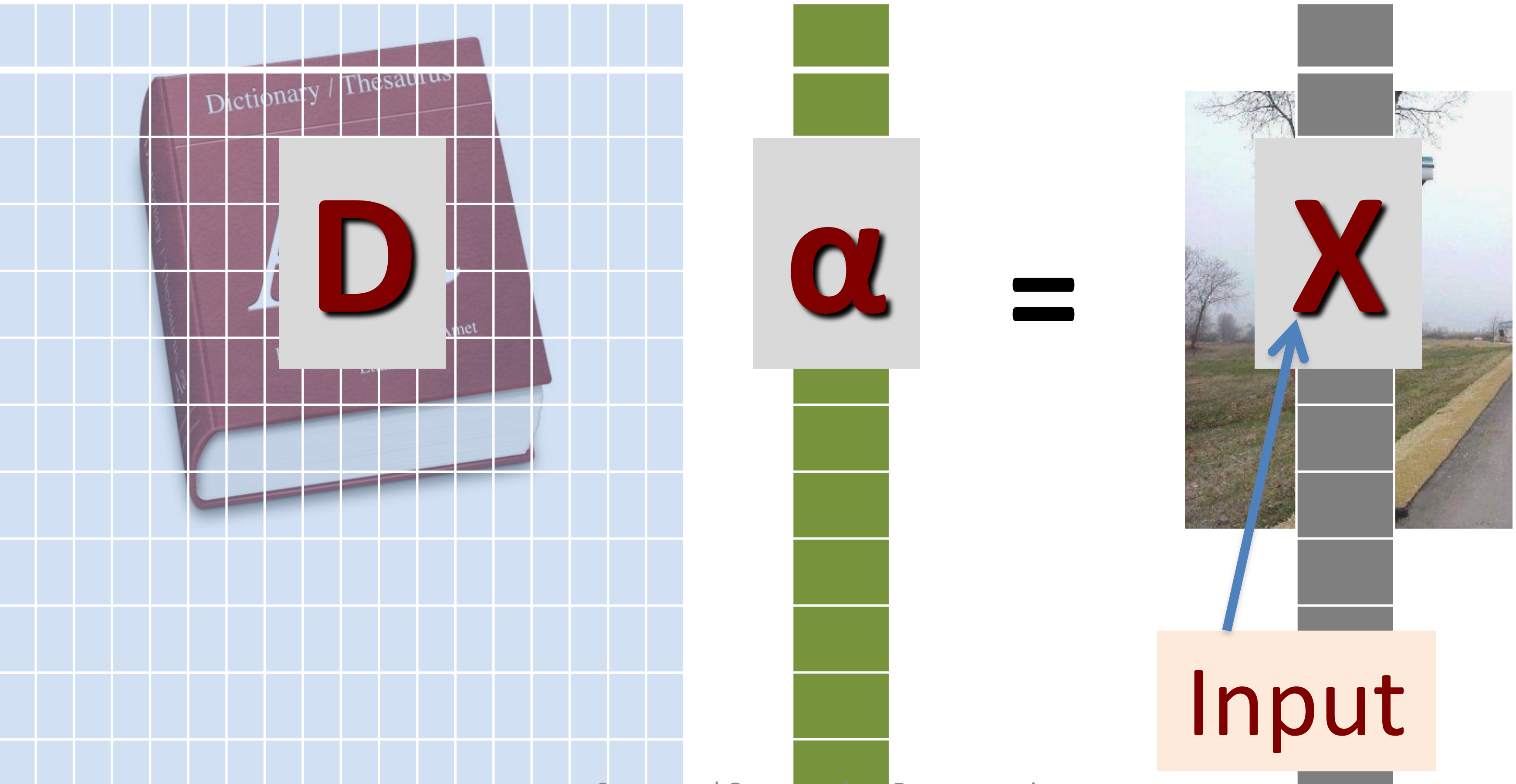


Why Dictionary-based Representations?

- 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 in an hour..

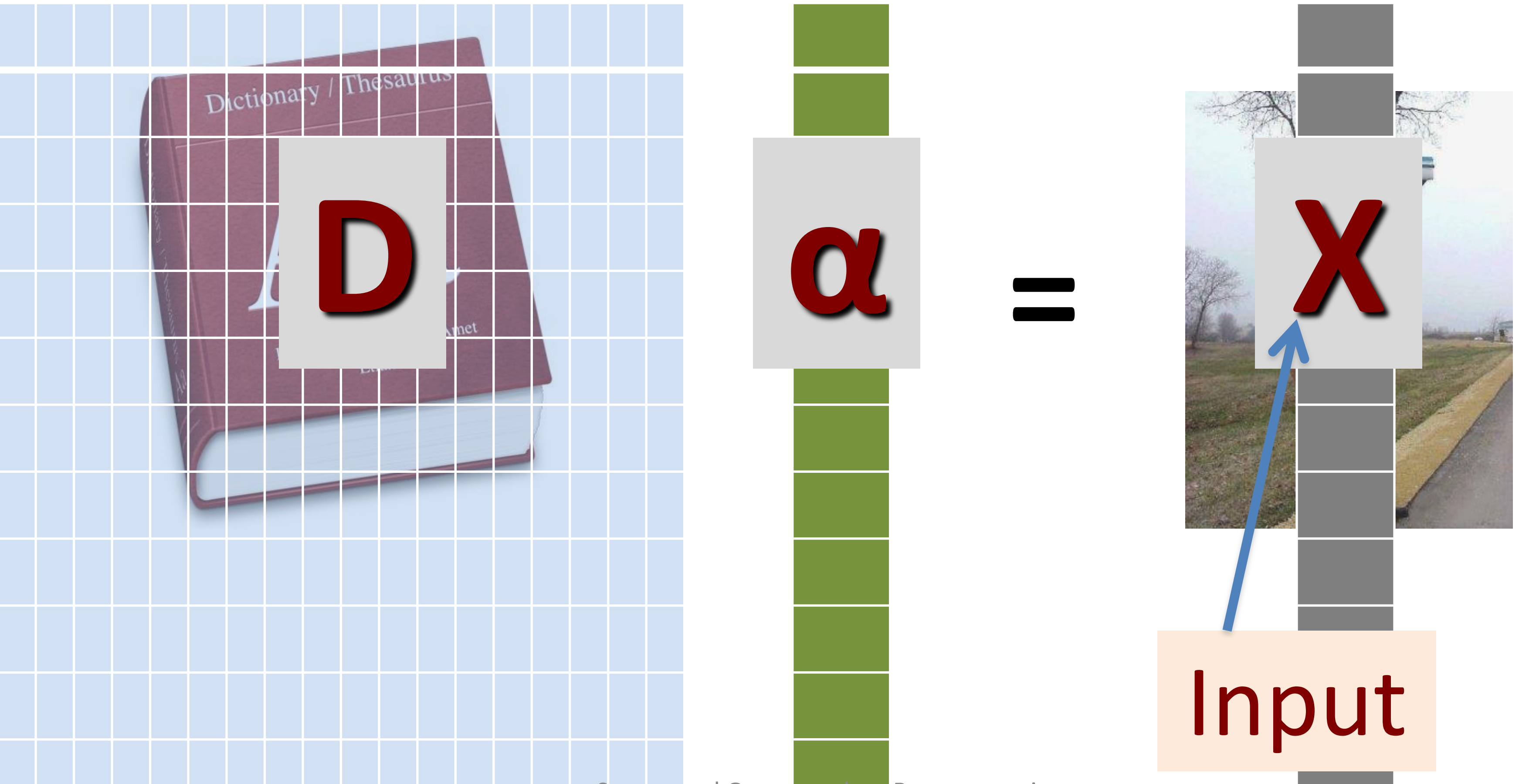
Problems

- 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, #(Basis Vectors)= #unknowns

$$\mathbf{D}.\boldsymbol{\alpha} = \mathbf{X}$$

Basis Vectors

Unknowns

Input data

Quick Linear Algebra Refresher

- Remember, #(Basis Vectors)= #unknowns

$$D.\alpha = X$$

Basis Vectors

Unknowns

Input data

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

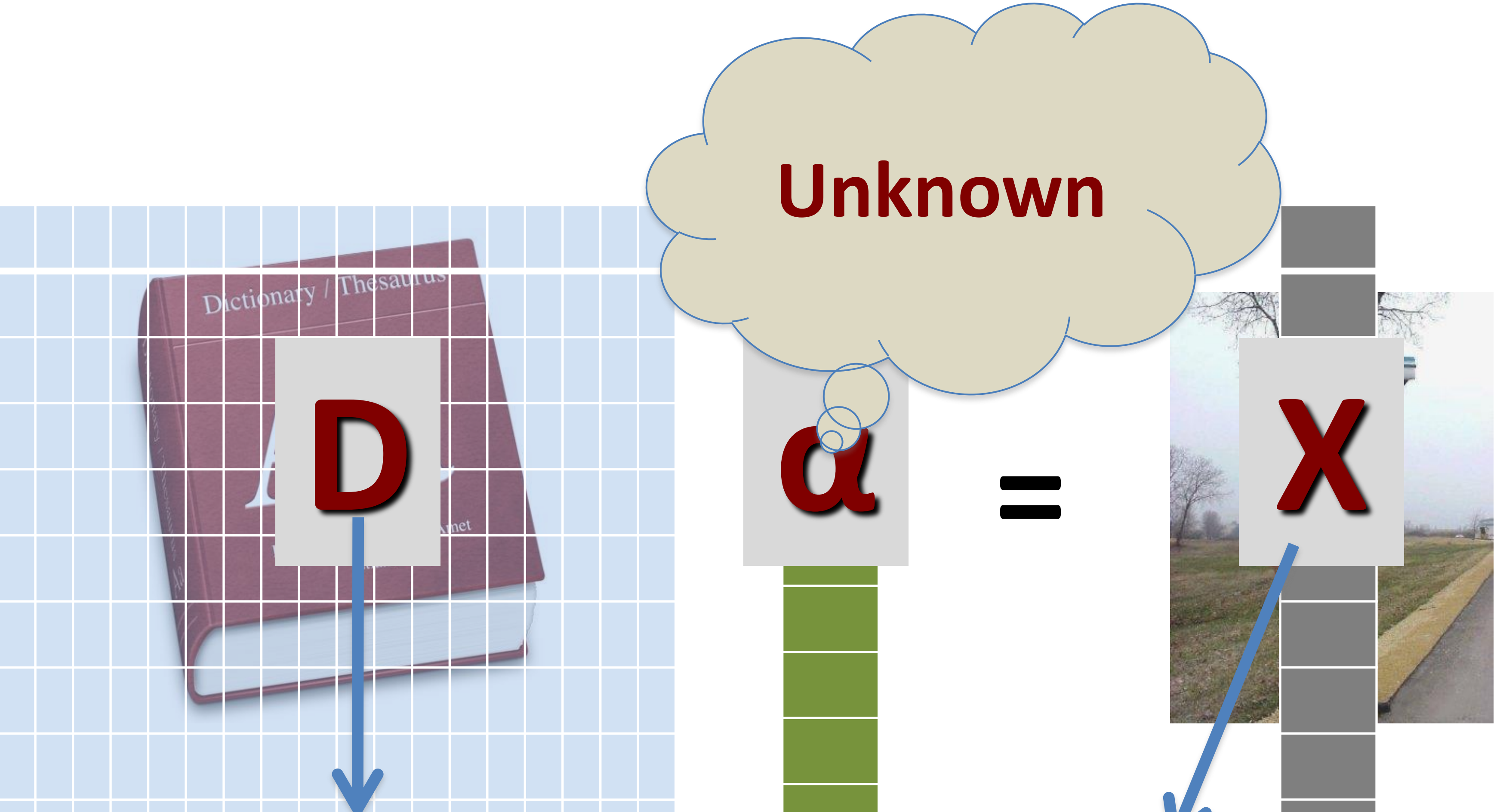
Quick Linear Algebra Refresher

$$D.\alpha = X$$

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

Our Case

Overcomplete Representation



#(Basis Vectors) > dimensions of the input

Overcompleteness and Sparsity

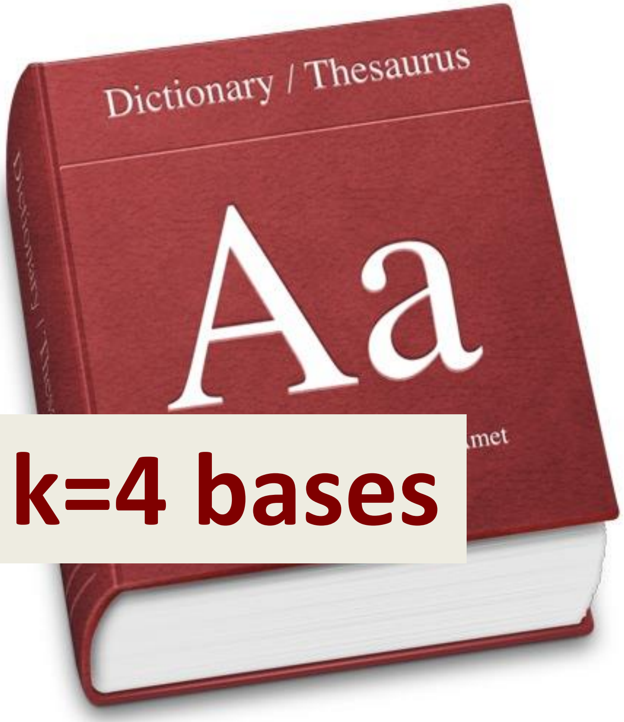
- To solve an overcomplete system of the type:

$$\mathbf{D}.\alpha = \mathbf{X}$$

- 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 ≤ 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

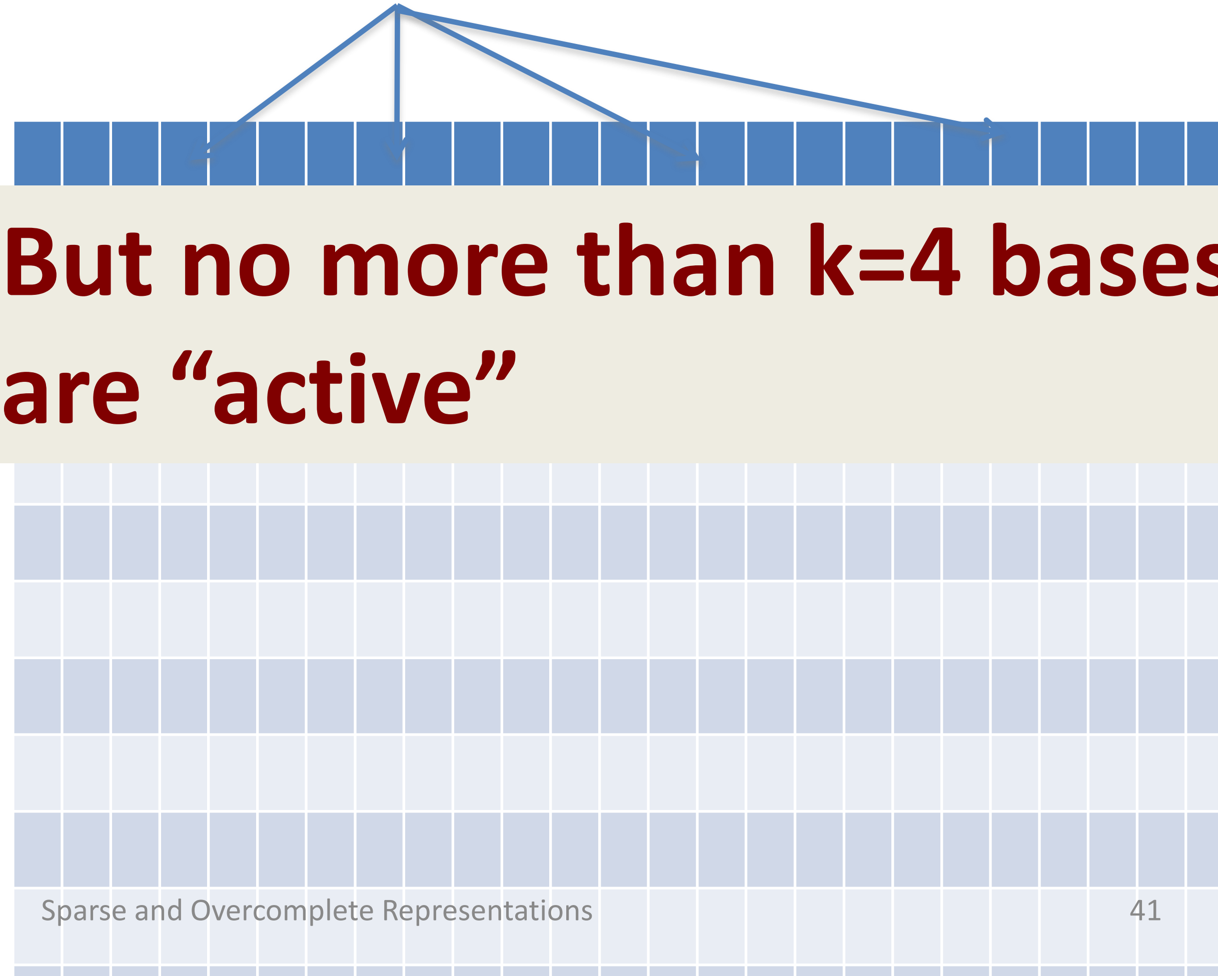
Using bases that we know…



But no more than k=4 bases

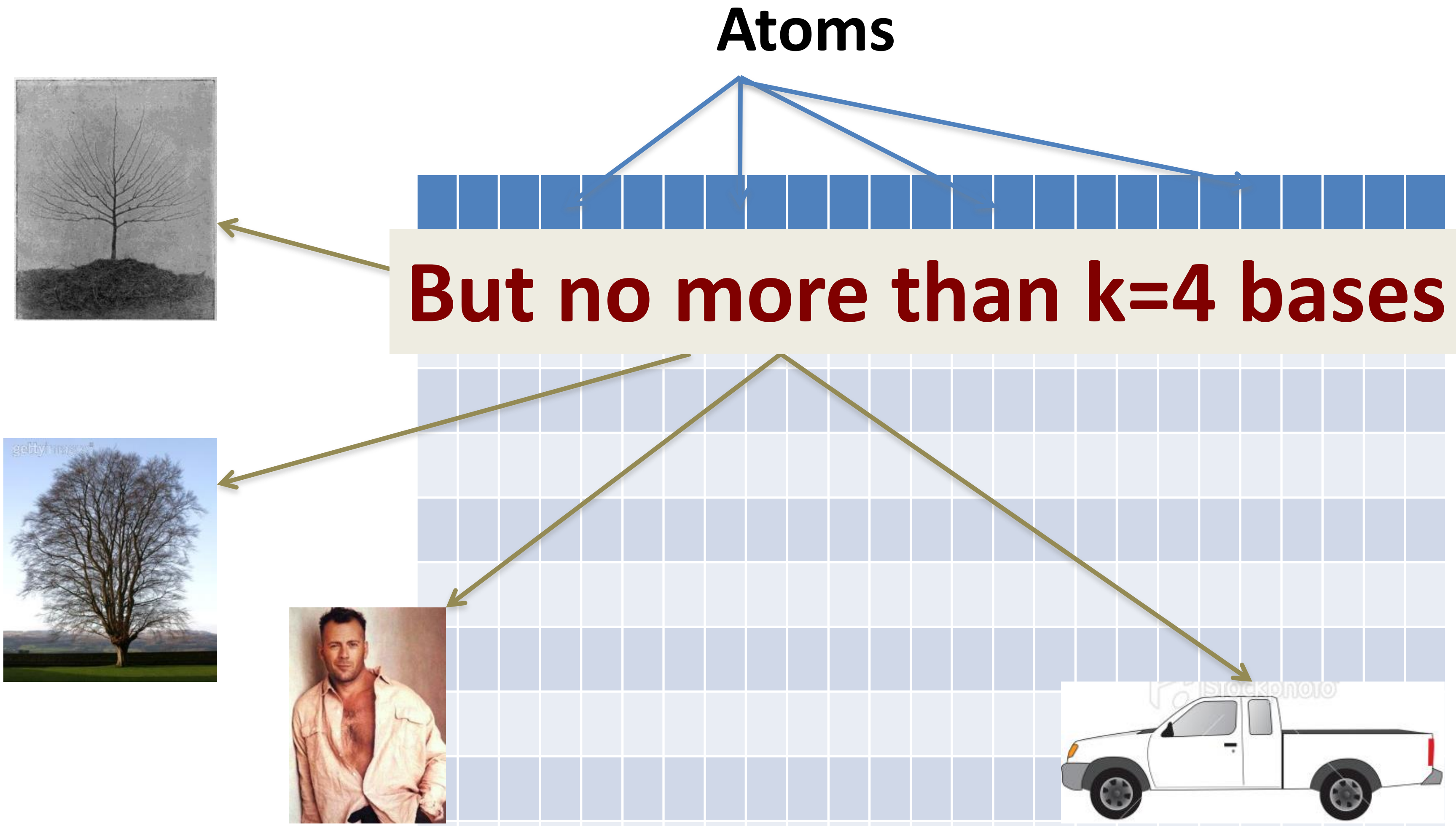
Overcompleteness and Sparsity

Atoms

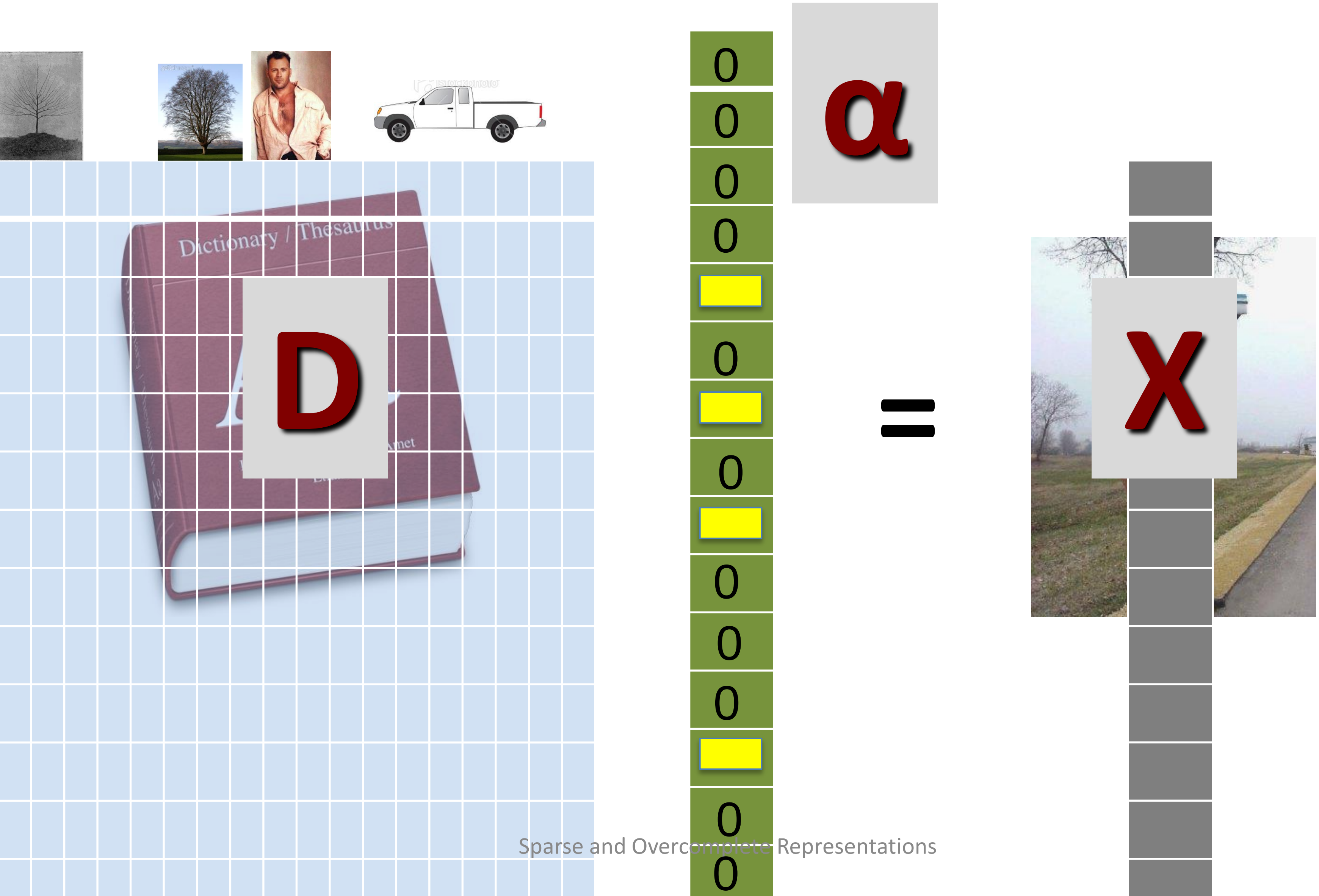


But no more than k=4 bases are "active"

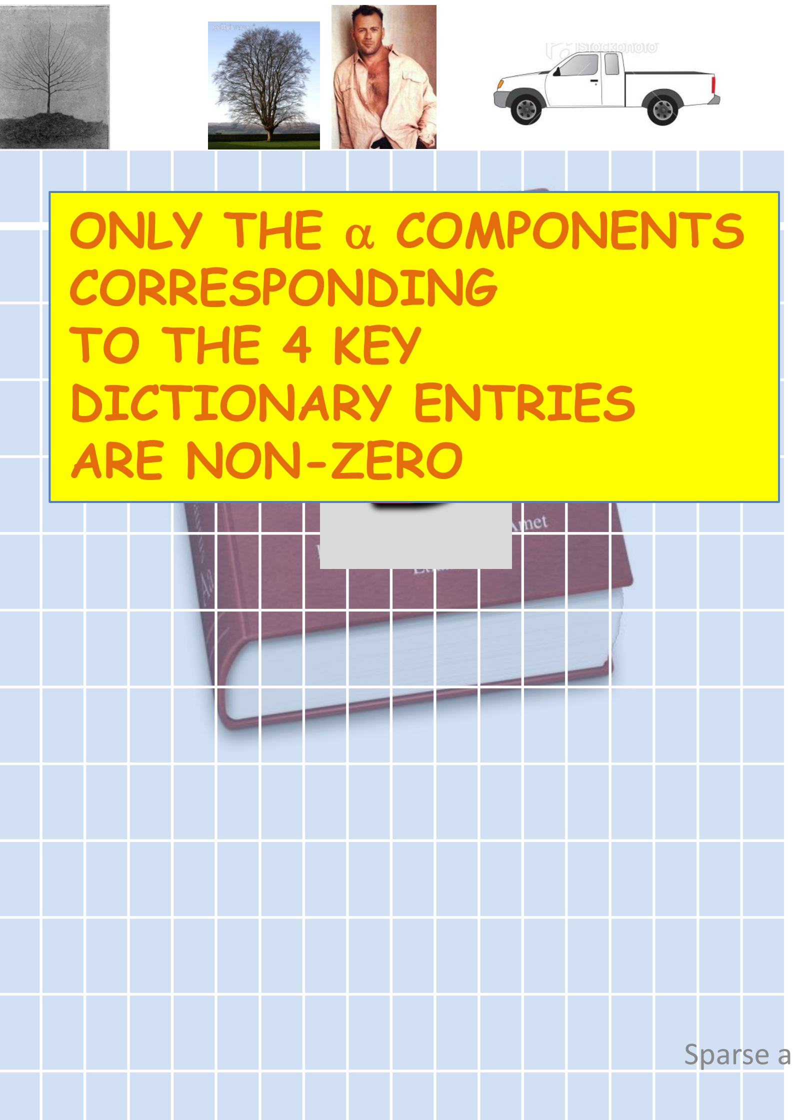
Overcompleteness and Sparsity

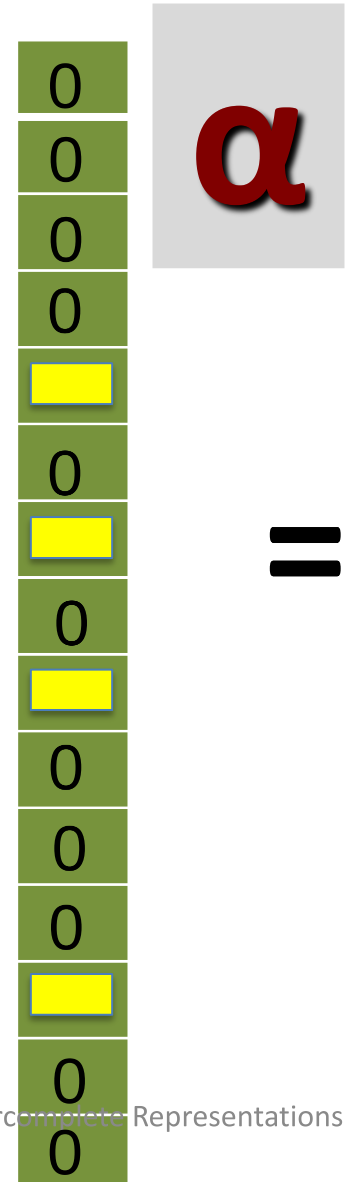


No more than 4 bases

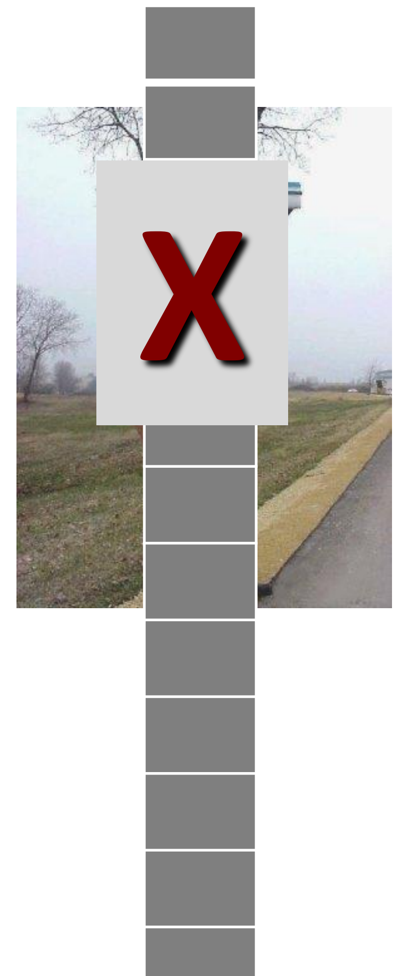


No more than 4 bases

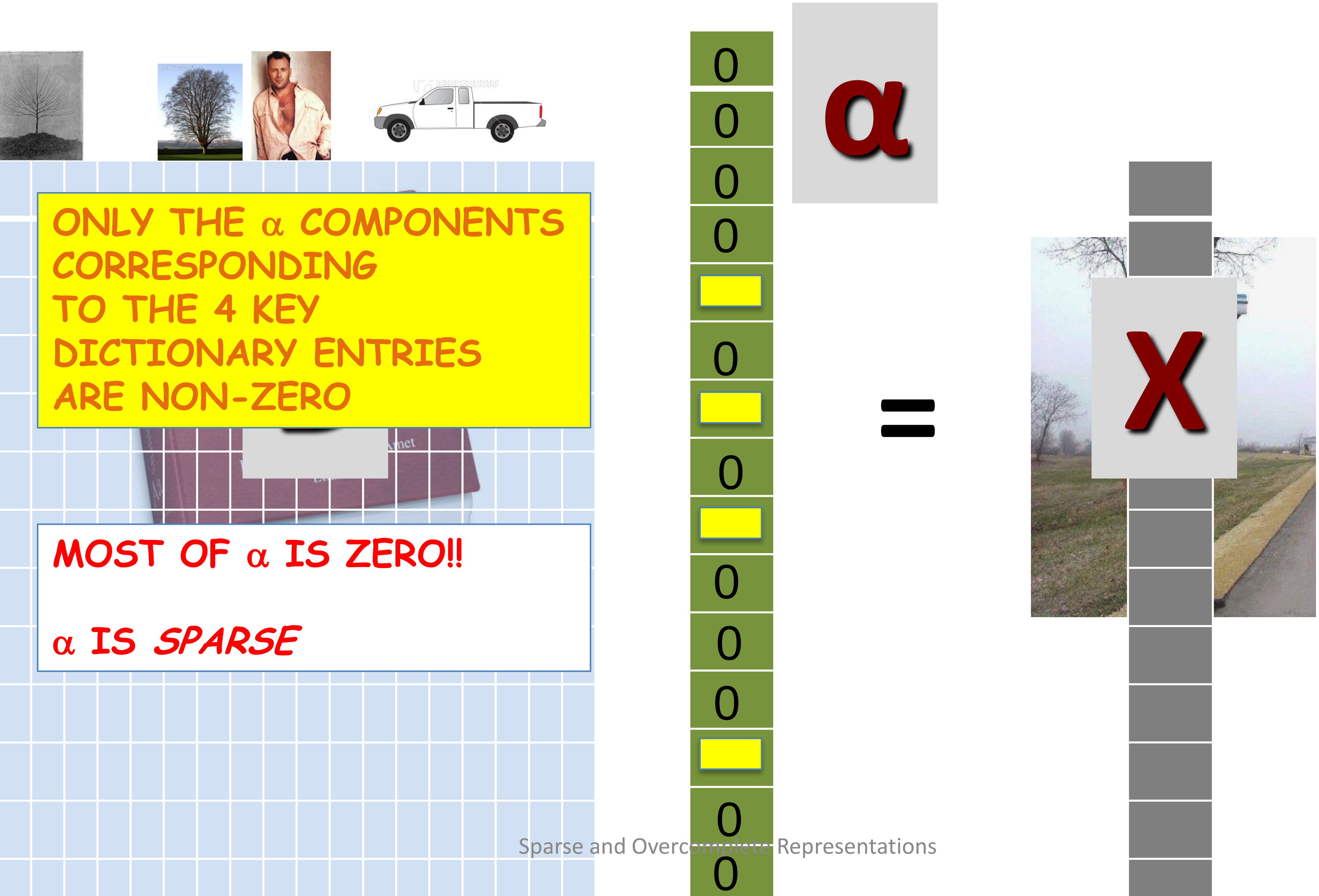




=



No more than 4 bases



ONLY THE α COMPONENTS CORRESPONDING TO THE 4 KEY DICTIONARY ENTRIES ARE NON-ZERO

MOST OF α IS ZERO!!

α IS *SPARSE*

Sparsity- Definition

- *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 α that generated it?

The Sparsity Problem

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

$$\underset{\underline{\alpha}}{Min} \ \left\|\underline{\alpha}\right\|_0$$

$$s.t. \ \ \underline{X} = \mathbf{D}\underline{\alpha}$$

The Sparsity Problem

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

$$\underset{\underline{\alpha}}{Min} \ \|\underline{\alpha}\|_0$$

$$s.t. \ \ \underline{X} = \mathbf{D}\underline{\alpha}$$

Counts the number of non-zero elements in α

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

$$\boxed{\begin{aligned} &\underset{\underline{\alpha}}{Min} \quad \left\|\underline{\alpha}\right\|_0 \\ &s.t. \quad \underline{X} = \mathbf{D}\underline{\alpha} \end{aligned}}$$

The Sparsity Problem

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

$$\boxed{\begin{array}{l} \underset{\underline{\alpha}}{Min} \ \left\| \underline{\alpha} \right\|_0 \\ s.t. \ \ \underline{X} = \mathbf{D}\underline{\alpha} \end{array}}$$

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 $\rightarrow$

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,

norm(ResidualInputSignal) < threshold

Matching Pursuit

Algorithm Matching Pursuit

Input: Signal: $f(t)$.

Output: List of coefficients: (a_n, g_{γ_n}).

Initialization:

$Rf_1 \leftarrow f(t)$;

Repeat

find $g_{\gamma_n} \in D$ 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: $\|Rf_n\| < threshold$)

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

Matching Pursuit	Basis Pursuit
Hard thresholding	
(remember the equations)	
Greedy optimization at each step	
Weights obtained using greedy rules	

Basis Pursuit (BP)

- Remember,

$$\boxed{\begin{array}{l} \underset{\underline{\alpha}}{Min} \quad \left\| \underline{\alpha} \right\|_0 \\ s.t. \quad \underline{X} = \mathbf{D}\underline{\alpha} \end{array}}$$

Basis Pursuit

- Remember,

$$\boxed{\begin{array}{l} \underset{\underline{\alpha}}{Min} \; \left\| \underline{\alpha} \right\|_0 \\ s.t. \;\; \underline{X} = \mathbf{D}\underline{\alpha} \end{array}}$$

In the general case, this is intractable

Basis Pursuit

- Remember,

$$\boxed{\begin{aligned} &\underset{\underline{\alpha}}{Min}\ \left\|\underline{\alpha}\right\|_0 \\ &s.t.\ \ \underline{X} = \mathbf{D}\underline{\alpha} \end{aligned}}$$

In the general case, this is intractable

Requires combinatorial optimization

Basis Pursuit

- Replace the intractable expression by an expression that is solvable

$$\boxed{\begin{aligned} &\underset{\underline{\alpha}}{Min}\ \left\|\underline{\alpha}\right\|_1 \\ &s.t.\ \ \underline{X} = \mathbf{D}\underline{\alpha} \end{aligned}}$$

Basis Pursuit

- Replace the intractable expression by an expression that is solvable

$$\boxed{\begin{aligned} &\underset{\underline{\alpha}}{Min}\ \|\underline{\alpha}\|_1 \\ &s.t.\ \ \underline{X} = \mathbf{D}\underline{\alpha} \end{aligned}}$$

This will provide identical solutions when $\mathbf{D}$ obeys the ***Restricted Isometry Property***.

Basis Pursuit

- Replace the intractable expression by an expression that is solvable

$$\boxed{\begin{aligned} &\underset{\underline{\alpha}}{Min} \ \left\|\underline{\alpha}\right\|_1 \\ &s.t. \ \ \underline{X} = \mathbf{D}\underline{\alpha} \end{aligned}}$$

Objective

Constraint

Basis Pursuit

- We can formulate the optimization term as:

$$\boxed{\underset{\underline{\alpha}}{Min}\ \{\|\underline{X}-\mathbf{D}\underline{\alpha}\|^2+\lambda\|\underline{\alpha}\|_1\}}$$

Constraint

Objective

Basis Pursuit

- We can formulate the optimization term as:

$$\boxed{\underset{\underline{\alpha}}{Min}\ \{\|\underline{X}-\mathbf{D}\underline{\alpha}\|^2+\lambda\|\underline{\alpha}\|_1\}}$$

λ 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

$$\underset{\underline{\alpha}}{Min} \ \{\|\underline{X} - \mathbf{D}\underline{\alpha}\|^2 + \lambda\|\underline{\alpha}\|_1\}$$

λ is a penalty term on the non-zero elements and promotes sparsity

Basis Pursuit

$$\underset{\underline{\alpha}}{Min}\{\|\underline{X}-\mathbf{D}\underline{\alpha}\|^2+\lambda\|\underline{\alpha}\|_1\}$$

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

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

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

$$\nabla_j\|\underline{X}-\mathbf{D}\underline{\alpha}\|^2\le\lambda, \quad \text{if } \alpha_j=0$$

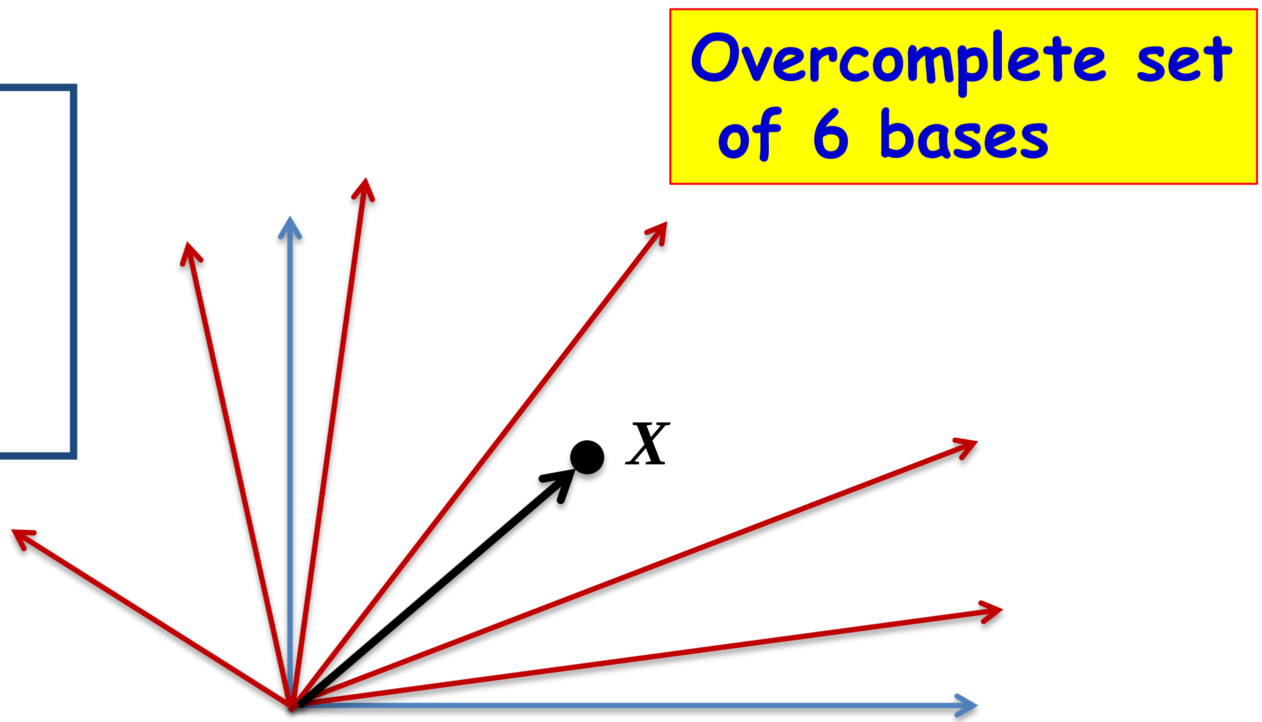
Basis Pursuit

- There are efficient ways to solve the LASSO formulation. [Link to Matlab code]
- Simplest solution: Coordinate descent algorithms
  - On webpage..

L_1 vs L_0

$$\underset{\underline{\alpha}}{Min} \|\underline{\alpha}\|_0$$

$$s.t. \underline{X} = \mathbf{D}\underline{\alpha}$$

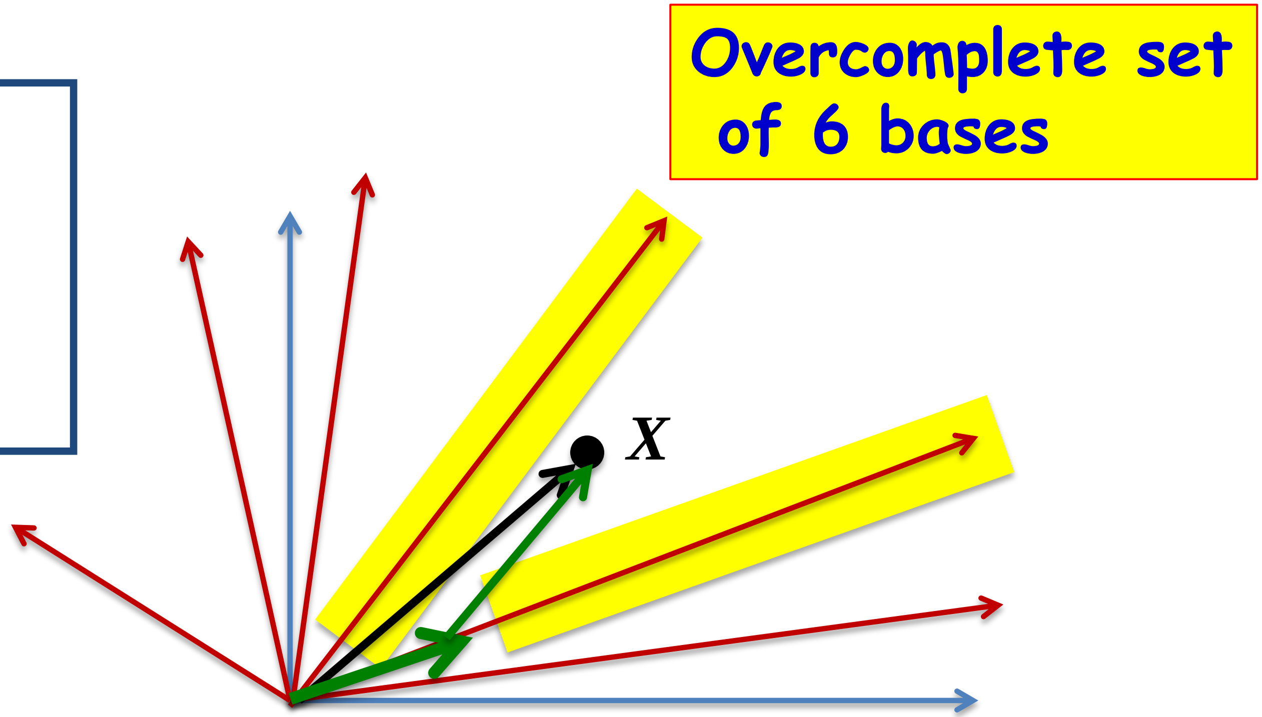


- **L_0 minimization**
  - Two-sparse solution
  - ANY pair of bases can explain $\boldsymbol{X}$ with 0 error

L_1 vs L_0

$$\underset{\underline{\alpha}}{Min} \left\| \underline{\alpha} \right\|_1$$

$$s.t. \underline{X} = \mathbf{D}\underline{\alpha}$$



- **L_1 minimization**
  – Two-sparse solution
  – All else being equal, the two closest bases are chosen

Comparing MP and BP

Matching Pursuit	Basis Pursuit
Hard thresholding	Soft thresholding
(remember the equations)	
Greedy optimization at each step	Global optimization
Weights obtained using greedy rules	Can force N-sparsity with appropriately chosen weights

General Formalisms

- L_0 minimization

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

- L_0 constrained optimization

$$\underset{\underline{\alpha}}{Min} \left\|\underline{X - \mathbf{D}\alpha}\right\|_2^2 \quad \underline{s.t. \|\underline{\alpha}\|_0 < C}$$

- L_1 minimization

$$\underset{\underline{\alpha}}{Min} \|\underline{\alpha}\|_1 \quad s.t.\ \underline{X} = \mathbf{D}\underline{\alpha}$$

- L_1 constrained optimization

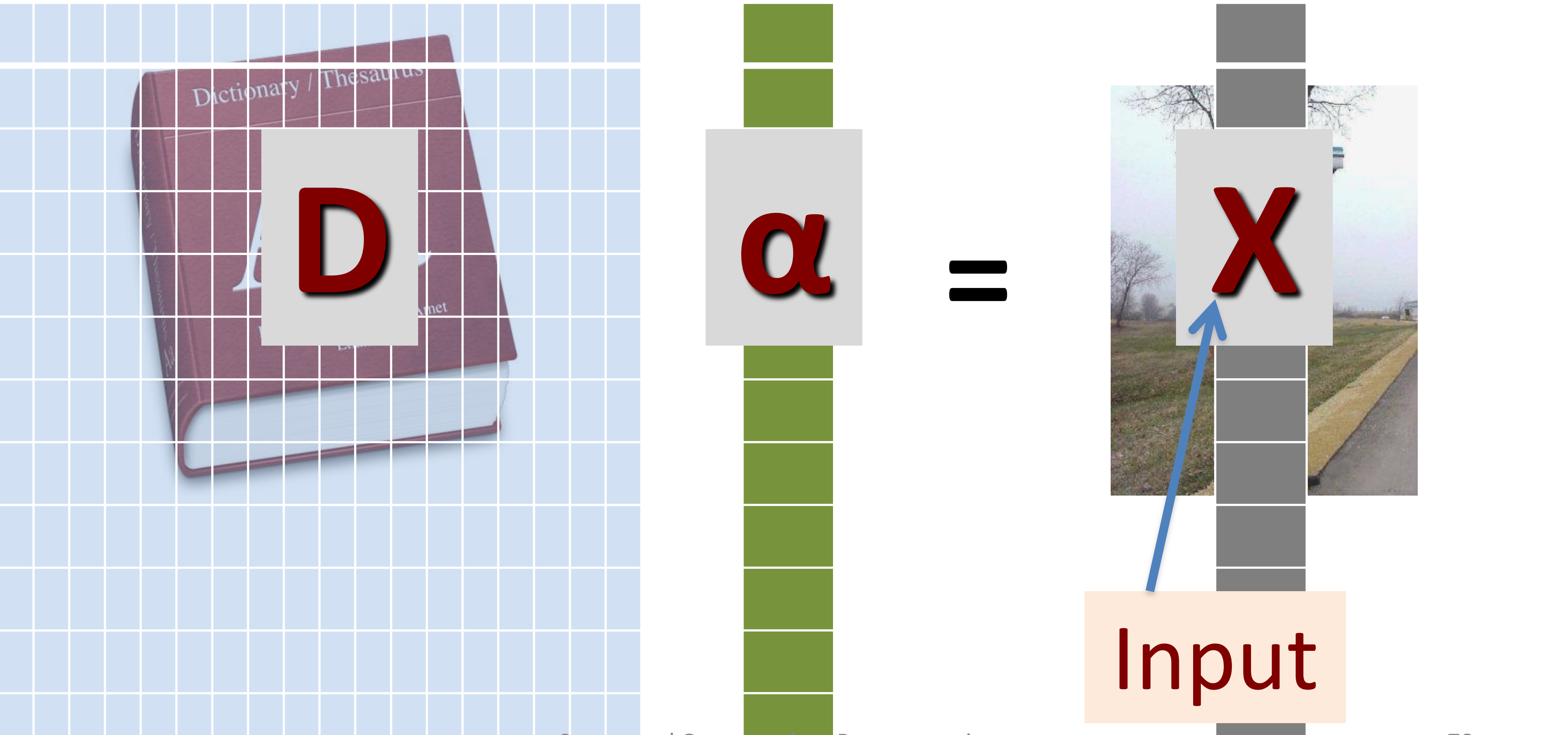
$$\underset{\underline{\alpha}}{Min} \left\|\underline{X - \mathbf{D}\alpha}\right\|_2^2 \quad \underline{s.t. \|\underline{\alpha}\|_1 < C}$$

Many Other Methods..

- Iterative Hard Thresholding (IHT)
- CoSAMP
- OMP
- …

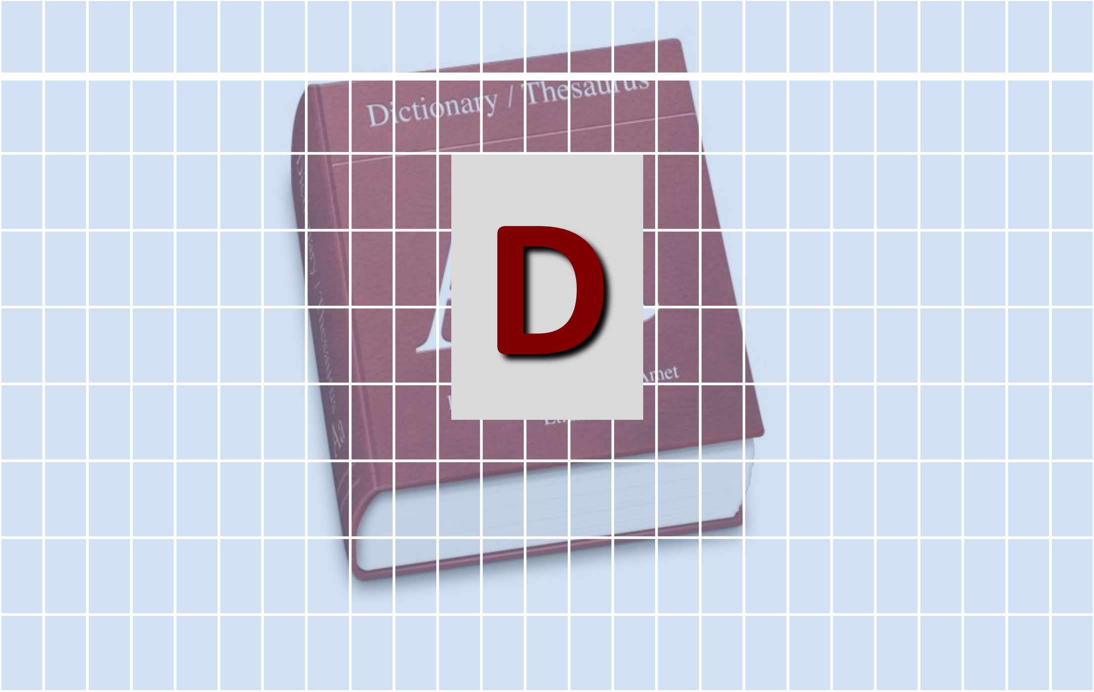
Problems

- How to obtain the dictionary
  - Which will give us meaningful representations
- How to compute the weights?



Dictionaries: Compressive Sensing

- Just random vectors!

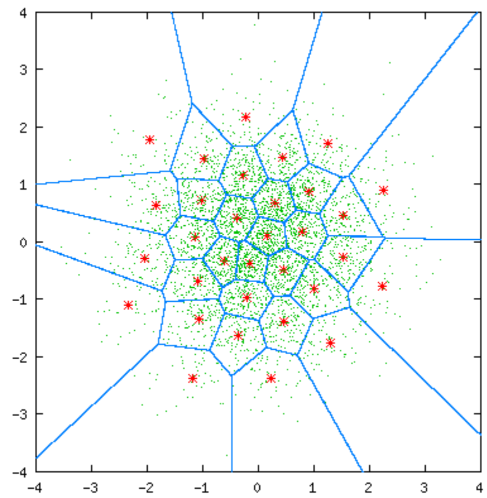


More Structured ways of Constructing Dictionaries

- 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



Train the codebook from training data using K-means

- 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

K-Means for Dictionaries

Each column is a codeword (centroid) from the codebook

D

α

$$\begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

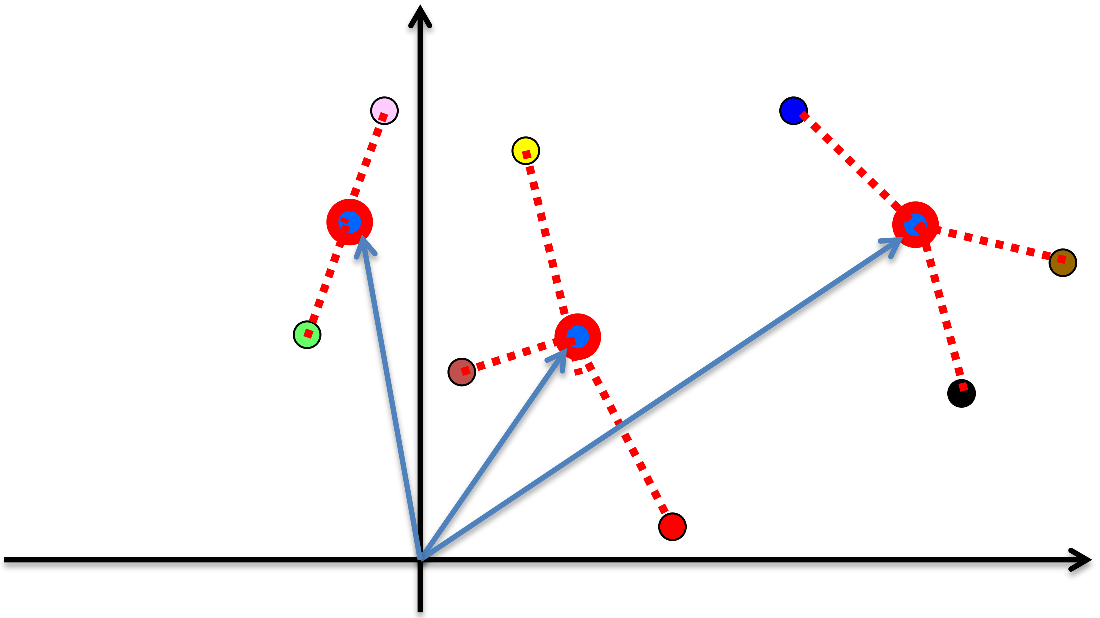
X

- α *must* be 1 sparse
- Only α entry must 1

$$|| \propto ||_0 = 1$$

$$|| \propto ||_1 = 1$$

K-Means



- Learn Codewords to minimize the total squared length of the training vectors from the closest codeword

Length-unconstrained
K-Means for Dictionaries

Each column is a codeword (centroid) from the codebook

D

α

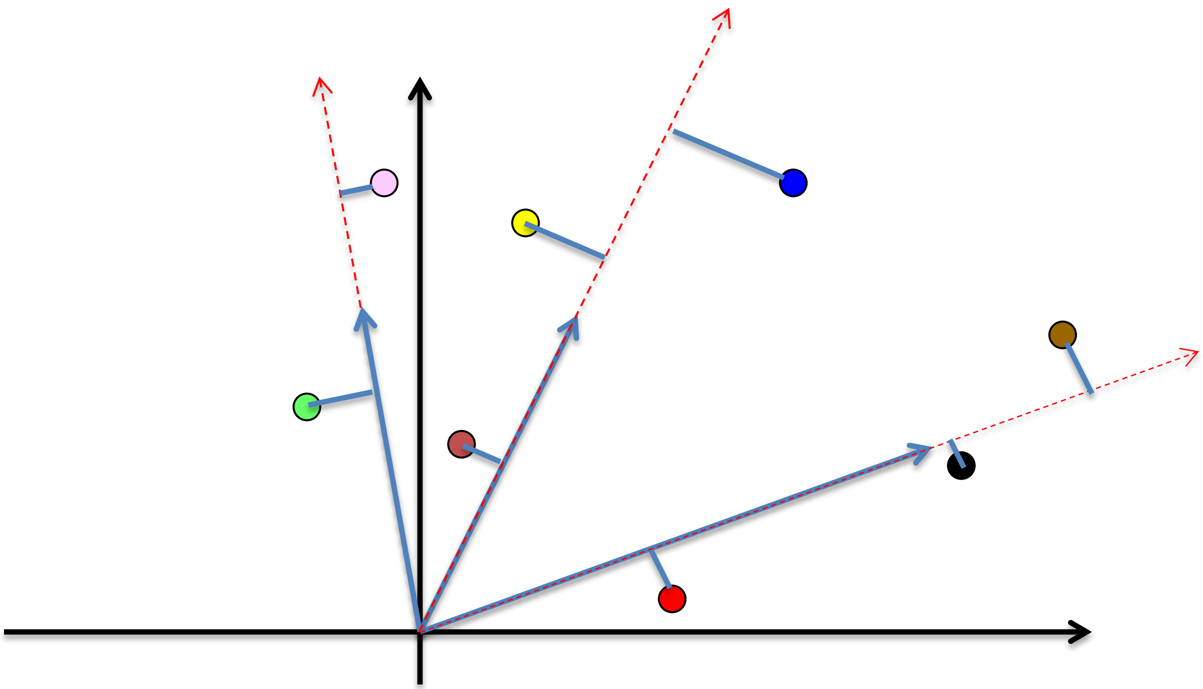
$$\begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 3 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

X

- α *must* be 1 sparse
- No restriction on α value

$$|| \propto ||_0 = 1$$

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

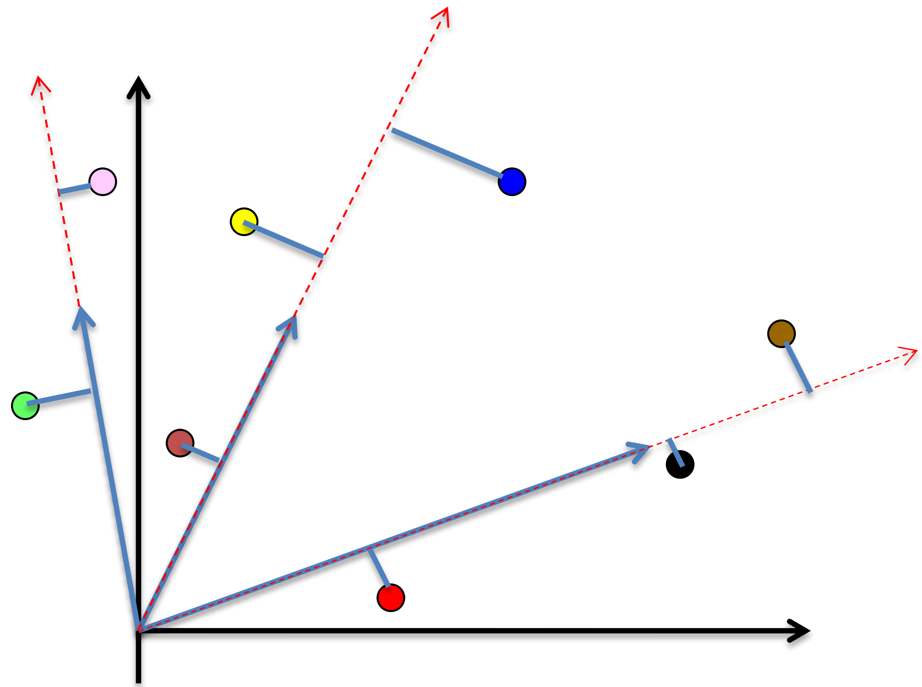
2. For each data point ***x***, find the projection from the centroid for each cluster

   - $$p_{cluster} = \left| x^T m_{cluster} \right|$$

3. Put data point in the cluster of the closest centroid

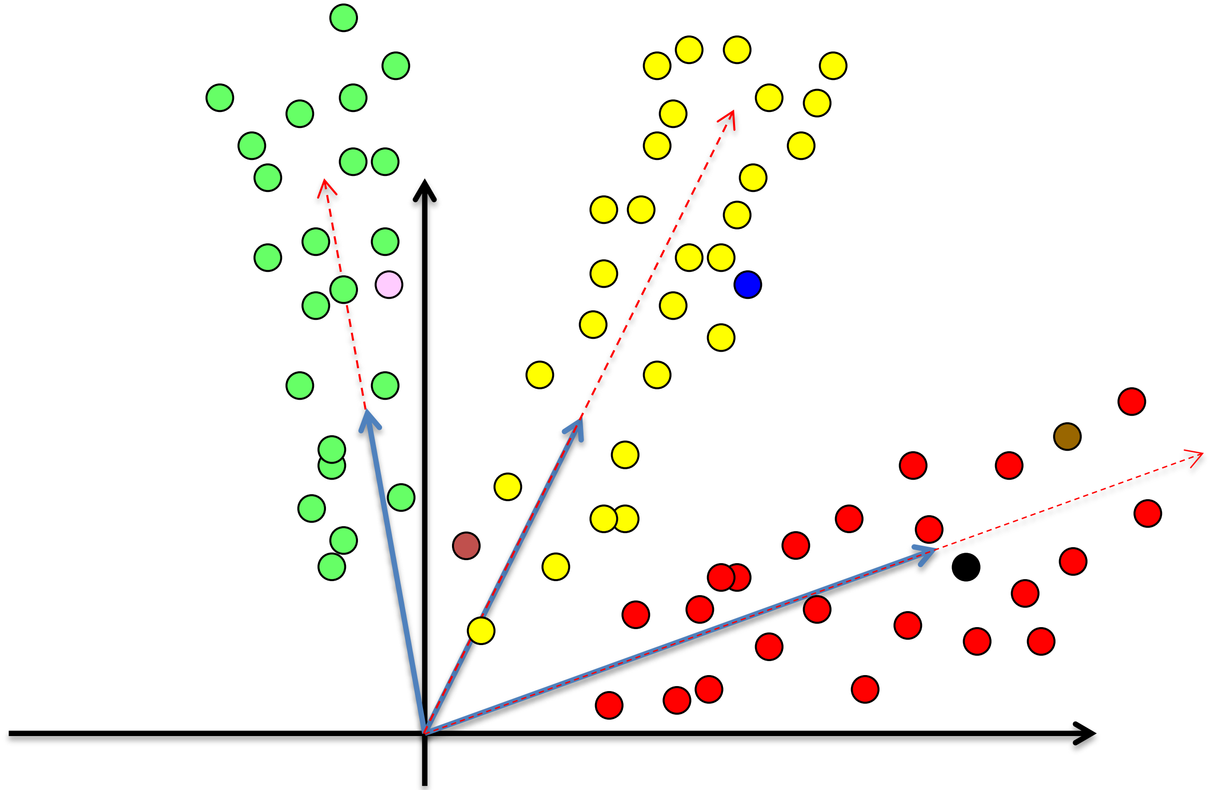
   - Cluster for which $\boldsymbol{p}_{\text{cluster}}$ is *maximum*

4. When all data points are clustered, recompute centroids



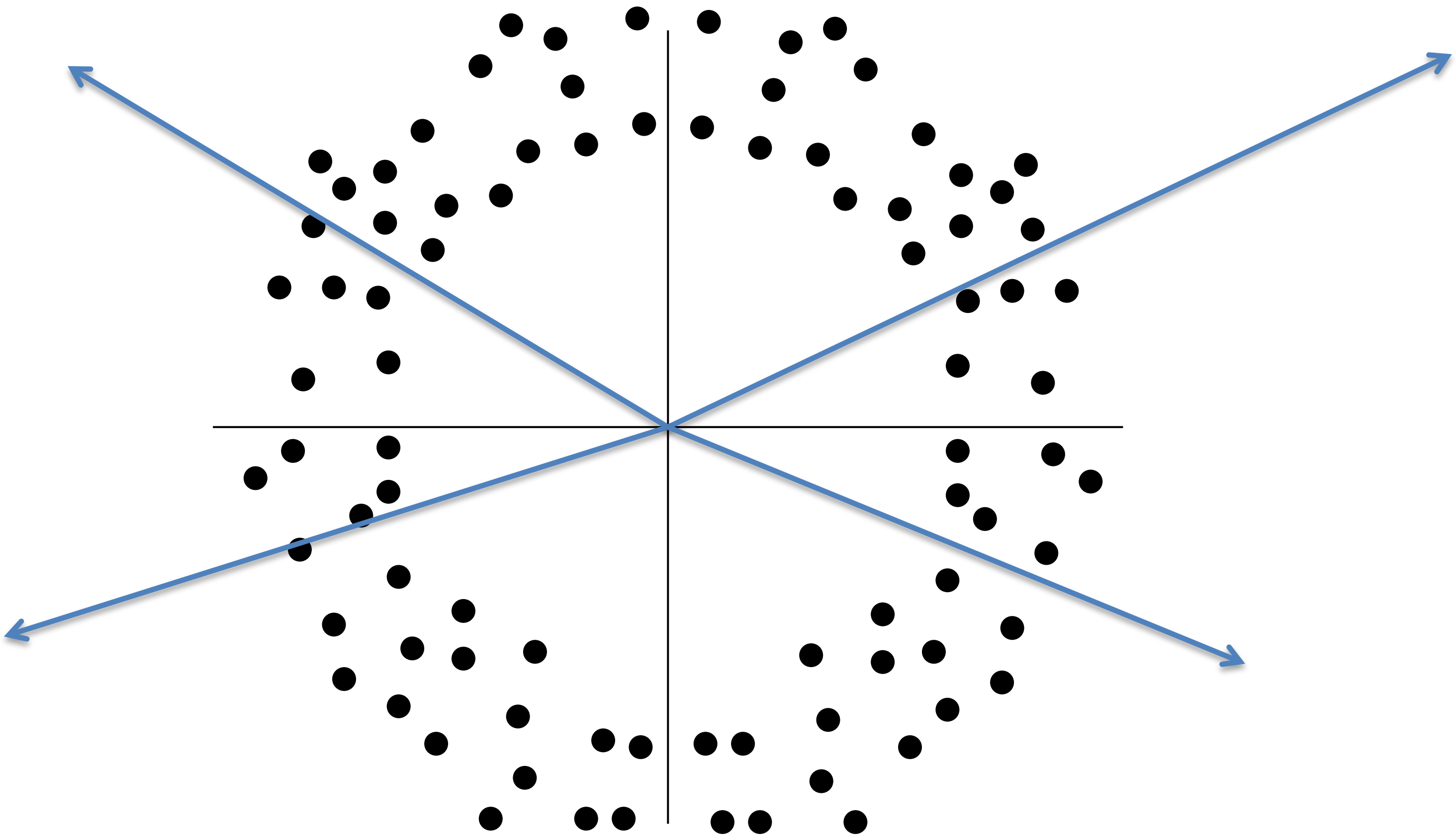
$$m_{cluster} = \text{Principal Eigenvector}(\{x \mid x \in cluster\})$$

Problem



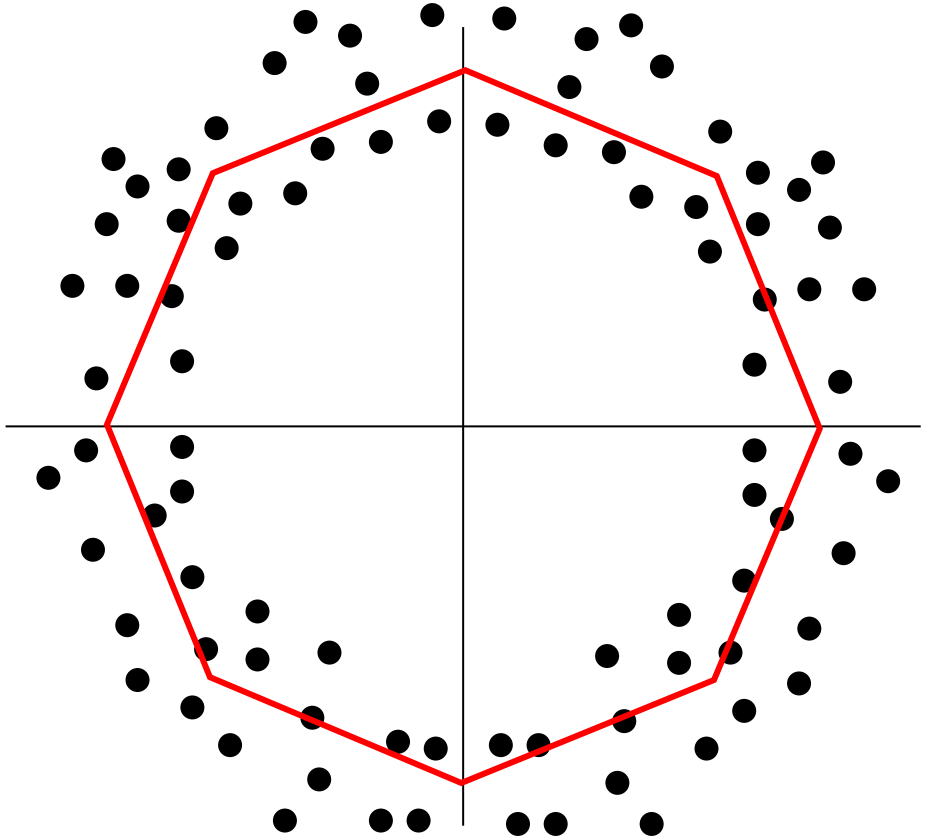
- Only represents *Radial* patterns

What about this pattern?



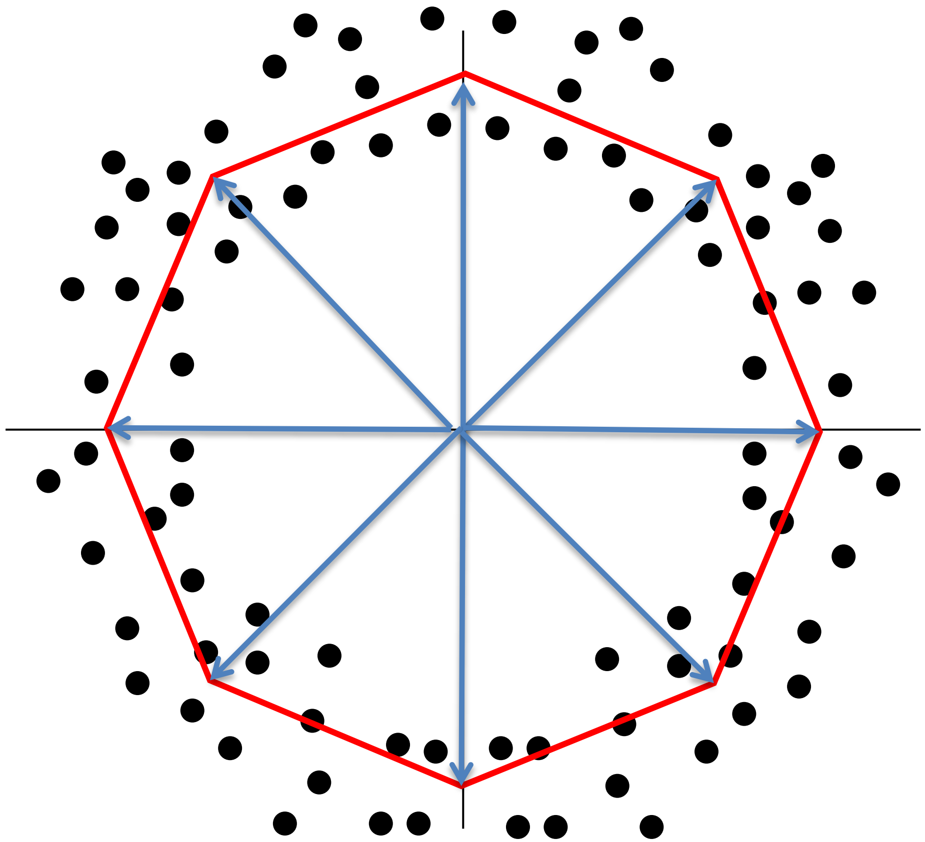
- Dictionary entries that represent radial patterns will not capture this structure
  - 1-sparse representations will not do

What about this pattern?



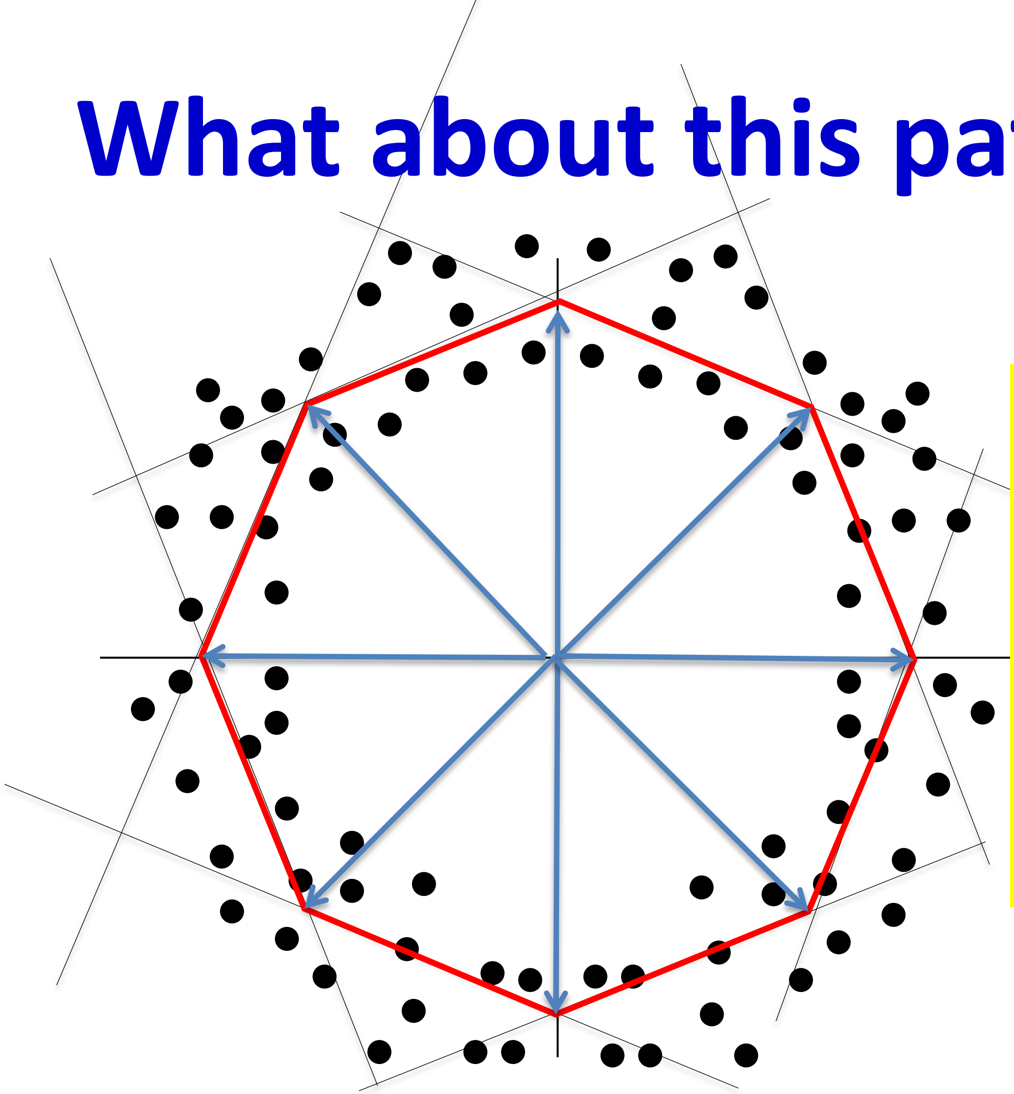
- We need ***AFFINE*** *patterns*

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?



Every line is a (constrined) combination of two bases

2-sparse

Constraint:
Line = $a.b_1 + (1-a)b_2$

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

Codebooks for K sparsity?

Each column is a codeword (centroid) from the codebook

D

α

0
0
0
0
3
0
1
0
0
4
0
0
0
0

X

- α *must* be *k* sparse
- No restriction on α value

$$||\propto||_0 = k$$

K SVD

- Initialize Codebook

$\mathrm{D} =$

1. For every vector, compute K-sparse alphas

$\alpha =$

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

K-SVD

2. For each codeword (k):

- For each vector $\boldsymbol{x}$
  - Subtract the contribution of all other codewords to obtain $\mathbf{e}_k(\boldsymbol{x})$
    - Codeword-specific residual
- Compute the principal Eigen vector of $\{\mathbf{e}_k(\boldsymbol{x})\}$

3. Return to step 1

$\mathbf{D}_j \quad \mathbf{j} \mathrel{!}= 1$

$\mathrm{D} =$

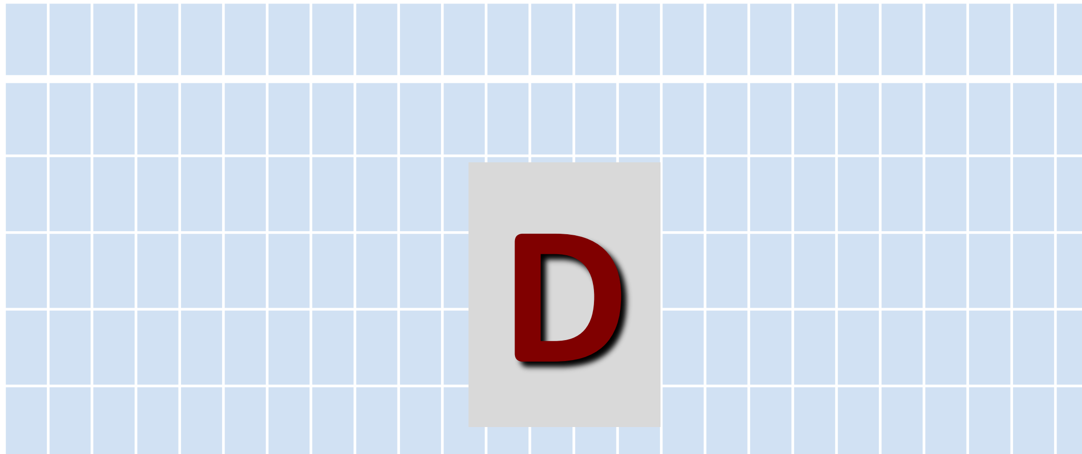
$\alpha =$

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

$\alpha_j(\mathbf{x}) \quad \mathbf{j} \mathrel{!}= 1$

$$e_k(x) = x - \sum_{j \neq k} \propto_j D_j$$

K-SVD

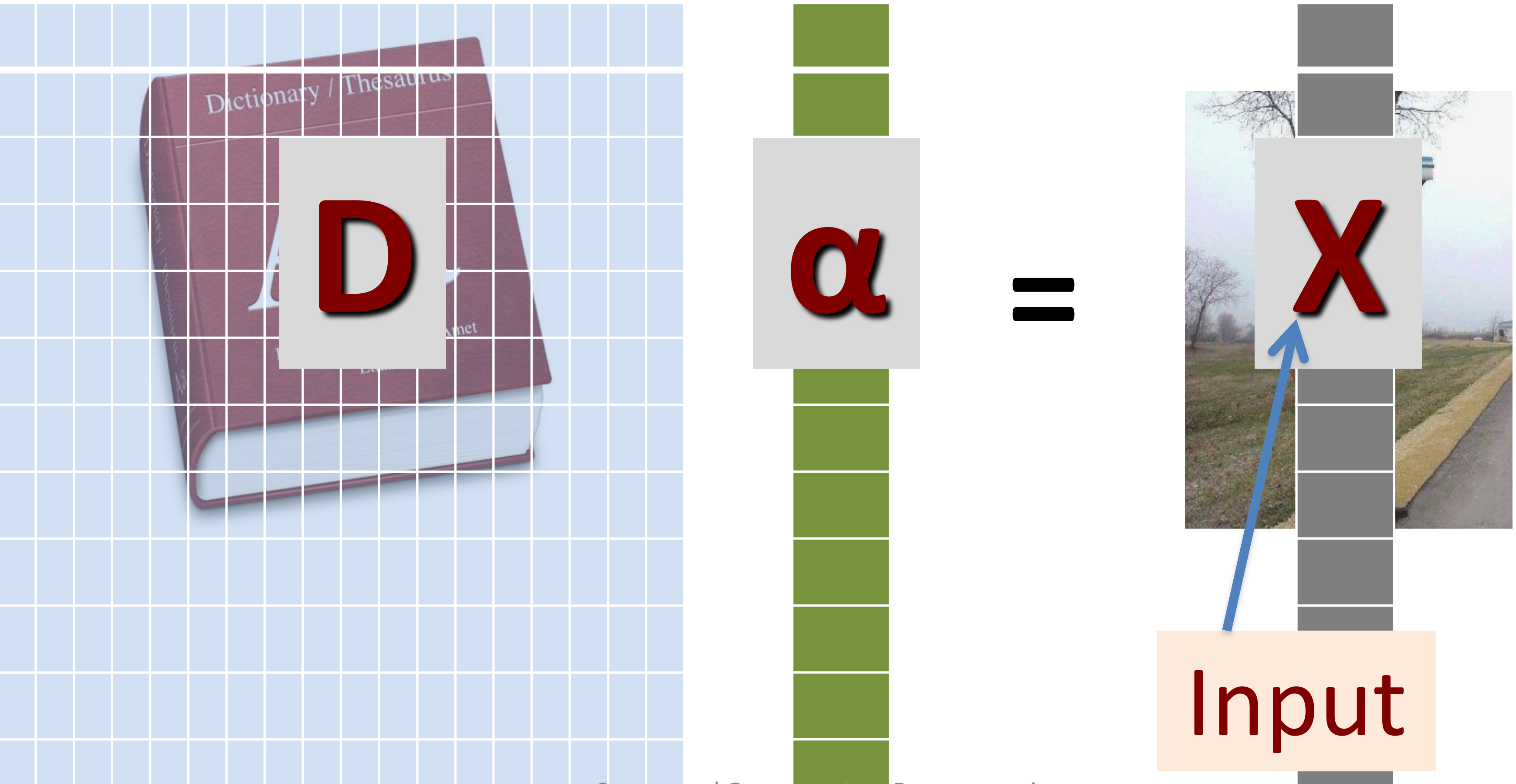


- Termination of each iteration: Updated dictionary

- 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

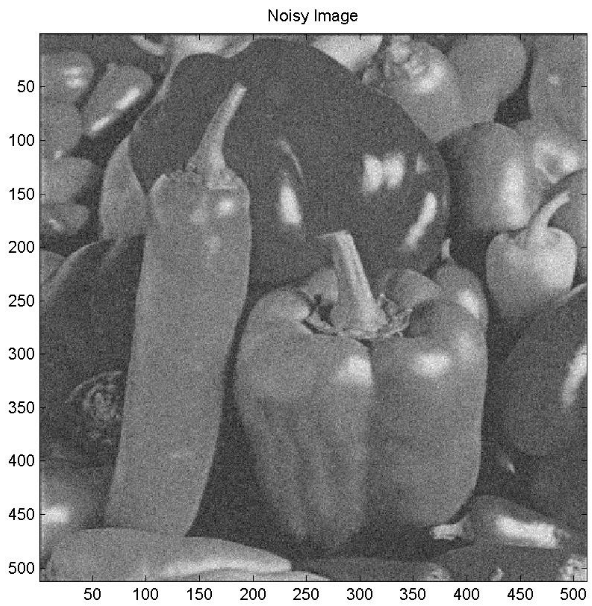
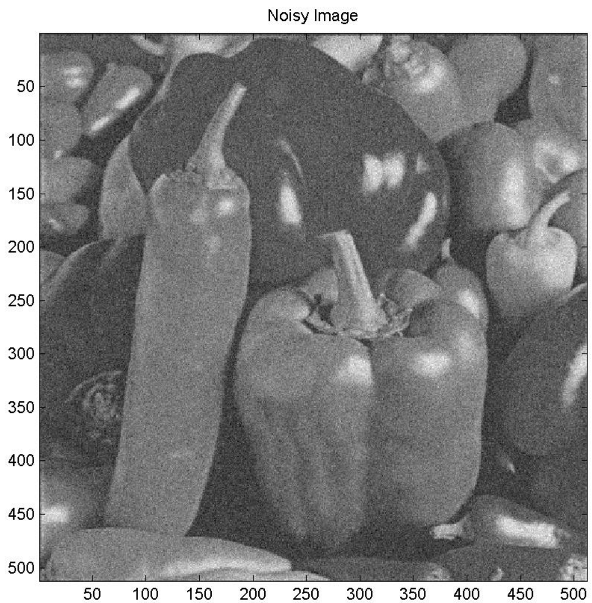


Image Denoising

- Here’s what we want



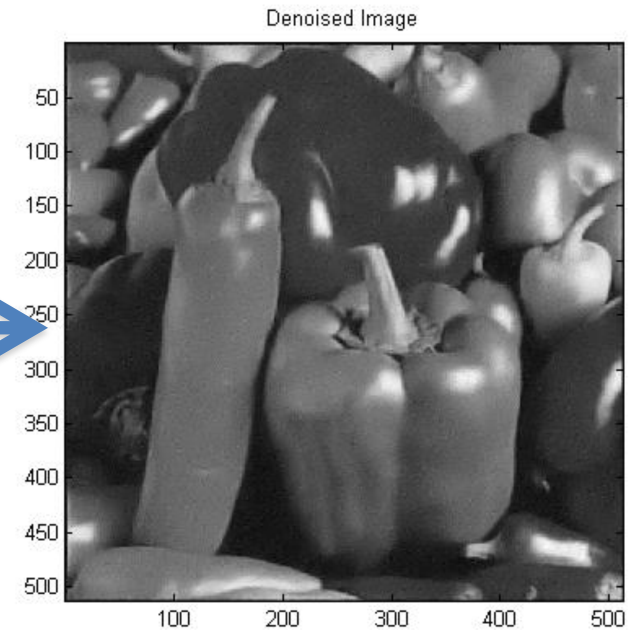


Image Denoising

- Here's what we want

The Image Denoising Problem

- Given an image
- Remove Gaussian additive noise from it

Image Denoising

Noisy Input

Orig. Image

$$Y = X + \varepsilon$$

Gaussian Noise

$$\varepsilon = N(0, \sigma)$$

Image Denoising

- Remove the noise from **Y**, to obtain **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 x 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 α so as to minimize the value of the equation below:

$$\underset{\underline{\alpha}}{Min}\ \{\|\underline{X} - \mathbf{D}\underline{\alpha}\|^2 + \lambda\|\underline{\alpha}\|_0\}$$

$$\underset{\underline{\alpha}}{Min}\{\|\underline{X} - \mathbf{D}\underline{\alpha}\|^2 + \lambda\|\underline{\alpha}\|_1\}$$

Image Denoising

$$\underset{\underline{\alpha}}{Min} \; \{ \| \underline{X} - \mathbf{D}\underline{\alpha} \|^2 + \lambda \| \underline{\alpha} \|_1 \}$$

- In the above, *X* is a patch.

Image Denoising

$$\underset{\underline{\alpha}}{Min}\ \{\|\underline{X}-\mathbf{D}\underline{\alpha}\|^2+\lambda\|\underline{\alpha}\|_1\}$$

- 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

$$\underset{\underline{\alpha_{ij}},X}{Min}\ \{\mu\left\|\underline{X}-Y\right\|_2^2+\sum_{ij}\left\|\underline{R_{ij}X}-\mathbf{D}\underline{\alpha}_{ij}\right\|_2^2$$

$$+\sum_{ij}\lambda_{ij}\left\|\underline{\alpha_{ij}}\right\|_0\}$$

Image Denoising

$$\underset{\underline{\alpha}_{ij},X}{Min} \left\{ \mu \left\| \underline{X} - Y \right\|_2^2 + \sum_{ij} \left\| \underline{R_{ij} X} - \mathbf{D} \underline{\alpha}_{ij} \right\|_2^2 + \sum_{ij} \lambda_{ij} \left\| \underline{\alpha_{ij}} \right\|_0 \right\}$$

(X – Y) is the error between the input and denoised image. μ is a penalty on the error.

Image Denoising

$$\underset{\underline{\alpha_{ij}, X}}{Min} \ \{\mu\|\underline{X}-Y\|_2^2+\sum_{ij}\|\underline{R_{ij}X}-\mathbf{D}\underline{\alpha}_{ij}\|_2^2$$

$$+\sum_{ij}\lambda_{ij}\|\underline{\alpha_{ij}}\|_0\}$$

Error bounding in each patch

-R_{ij} selects the $(ij)^{th}$ patch

-Terms in summation = no. of patches

Image Denoising

$$\underset{\underline{\alpha_{ij}},X}{Min}\ \{\mu\left\|\underline{X}-Y\right\|_2^2+\sum_{ij}\left\|\underline{R_{ij}X}-\mathbf{D}\underline{\alpha}_{ij}\right\|_2^2$$

$$+\sum_{ij}\lambda_{ij}\left\|\underline{\alpha_{ij}}\right\|_0\}$$

λ forces sparsity

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.

$$\underset{\underline{D,\alpha_{ij},X}}{Min}\ \{\mu\|\underline{X}-Y\|_2^2+\sum_{ij}\|\underline{R_{ij}X}-\mathbf{D}\underline{\alpha}_{ij}\|_2^2$$

$$+\sum_{ij}\lambda_{ij}\|\underline{\alpha_{ij}}\|_0\}$$

We can use the previous equation itself!!!

Image Denoising

$$\underline{\underset{D,\alpha_{ij},X}{Min}} \{\mu \left\| \underline{X} - Y \right\|_2^2 + \sum_{ij} \left\| \underline{R_{ij} X} - \mathbf{D} \underline{\alpha}_{ij} \right\|_2^2$$

$$+ \sum_{ij} \lambda_{ij} \left\| \underline{\alpha_{ij}} \right\|_0 \}$$

How do we estimate all 3 at once?

Image Denoising

$$\underset{\underline{D,\alpha_{ij},X}}{Min}\ \{\mu\|\underline{X}-Y\|_2^2+\sum_{ij}\|\underline{R_{ij}X}-\mathbf{D}\underline{\alpha}_{ij}\|_2^2$$

$$+\sum_{ij}\lambda_{ij}\|\underline{\alpha_{ij}}\|_0\}$$

How do we estimate all 3 at once?

We cannot estimate them at the same time!

Image Denoising

$$\underset{\underline{D,\alpha_{ij},X}}{Min} \; \{\mu\left\|\underline{X}-Y\right\|_2^2+\sum_{ij}\left\|\underline{R_{ij}X}-\mathbf{D}\underline{\alpha}_{ij}\right\|_2^2$$

$$+\sum_{ij}\lambda_{ij}\left\|\underline{\alpha_{ij}}\right\|_0\}$$

How do we estimate all 3 at once?

Fix 2, and find the optimal 3rd.

Image Denoising

$$\underset{\underline{D,\alpha_{ij},X}}{Min}\ \{\mu\left\|\underline{X}-Y\right\|_2^2+\sum_{ij}\left\|\underline{R_{ij}X}-\mathbf{D}\underline{\alpha}_{ij}\right\|_2^2$$

$$+\sum_{ij}\lambda_{ij}\left\|\underline{\alpha_{ij}}\right\|_0\}$$

Initialize X = Y

Image Denoising

$$\underset{\underline{\alpha}_{ij}}{Min}\ \{\mu\left\|\underline{X}-Y\right\|_2^2+\sum_{ij}\left\|\underline{R_{ij}X}-\mathbf{D}\underline{\alpha}_{ij}\right\|_2^2$$

$$+\sum_{ij}\lambda_{ij}\left\|\underline{\alpha_{ij}}\right\|_0\}$$

0

Initialize X = Y, initialize D

You know how to solve the remaining portion for α – MP, BP!

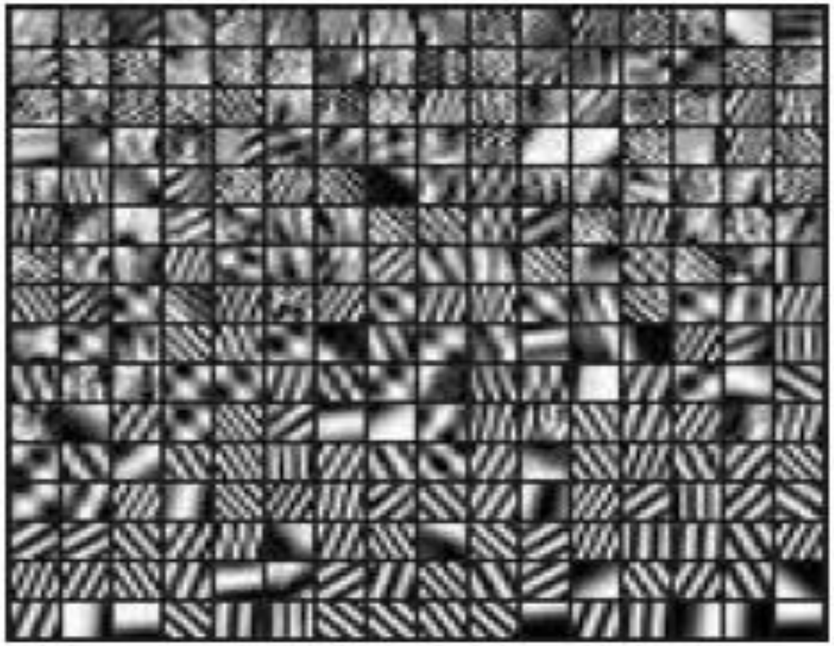
Image Denoising

- Now, update the dictionary D.
- Update D one column at a time, following the [K-SVD algorithm]
- K-SVD maintains the sparsity structure

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
- Iteratively update α and D

Image Denoising



Learned Dictionary for Face Image denoising

From: M. Elad and M. Aharon, *Image denoising via learned dictionaries and sparse representation*, CVPR, 2006.

Image Denoising

$$\underset{\underline{X}}{Min}\ \{\mu\|\underline{X}-Y\|_2^2+\sum_{ij}\|\underline{R_{ij}X}-\mathbf{D}\underline{\alpha}_{ij}\|_2^2$$

$$+\sum_{ij}\lambda_{ij}\|\underline{\alpha_{ij}}\|_0\}$$

Const. wrt X

We know D and α

The quadratic term above has a closed-form solution

Image Denoising

$$\underset{\underline{X}}{Min}\ \{\mu\|\underline{X}-Y\|_2^2+\sum_{ij}\|\underline{R_{ij}X}-\mathbf{D}\underline{\alpha}_{ij}\|_2^2$$

$$+\sum_{ij}\lambda_{ij}\|\underline{\alpha_{ij}}\|_0\}$$ Const. wrt X

We know D and α

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

Image Denoising

- Summarizing… We wanted to obtain 3 things

Image Denoising

- Summarizing… We wanted to obtain 3 things

- Weights **α**
- Dictionary **D**
- Denoised Image **X**

Image Denoising

- Summarizing… We wanted to obtain 3 things

➢ Weights **α – ** Your favorite pursuit algorithm
➢ Dictionary **D – ** Using K-SVD
➢ Denoised Image **X**

Image Denoising

- Summarizing… We wanted to obtain 3 things

➢ Weights **α** – Your favorite pursuit algorithm

➢ Dictionary **D** – Using K-SVD

➢ Denoised Image **X**

Iterating

Image Denoising

- Summarizing… We wanted to obtain 3 things

- ➢ Weights **α**
- ➢ Dictionary **D**
- ➢ Denoised Image **X-** Closed form solution

Image Denoising

- Here’s what we want

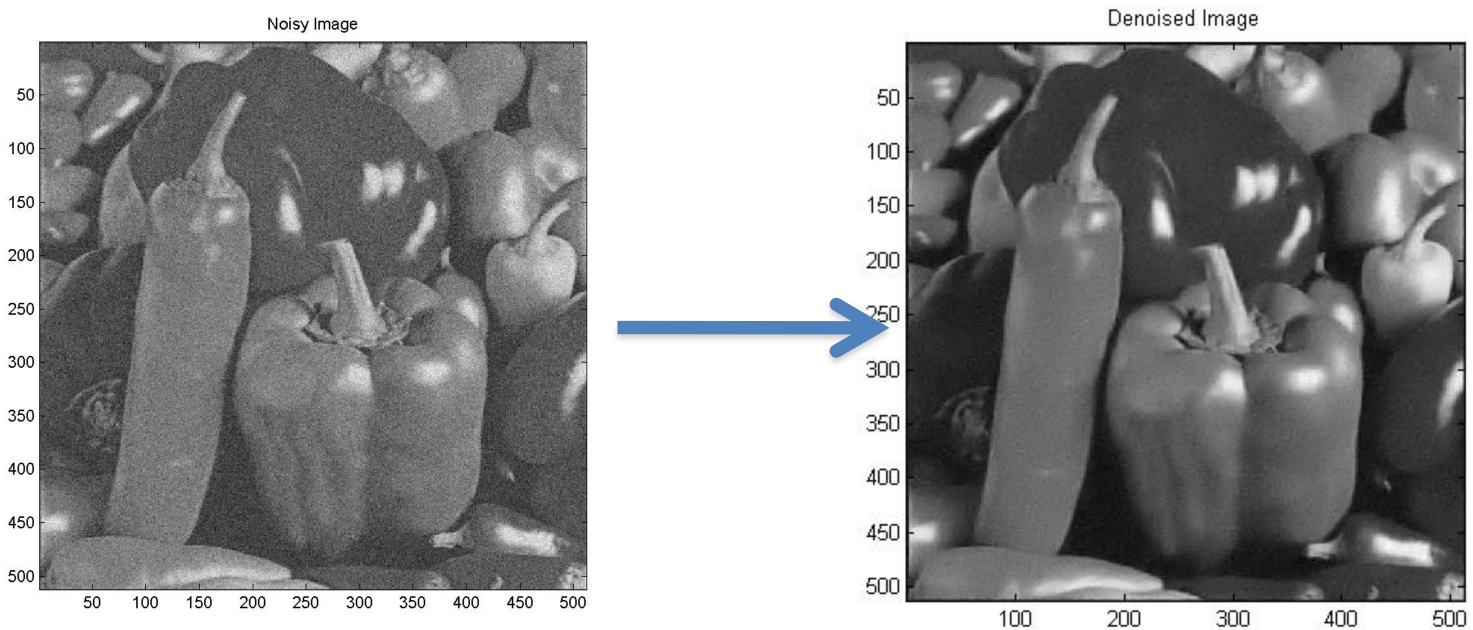
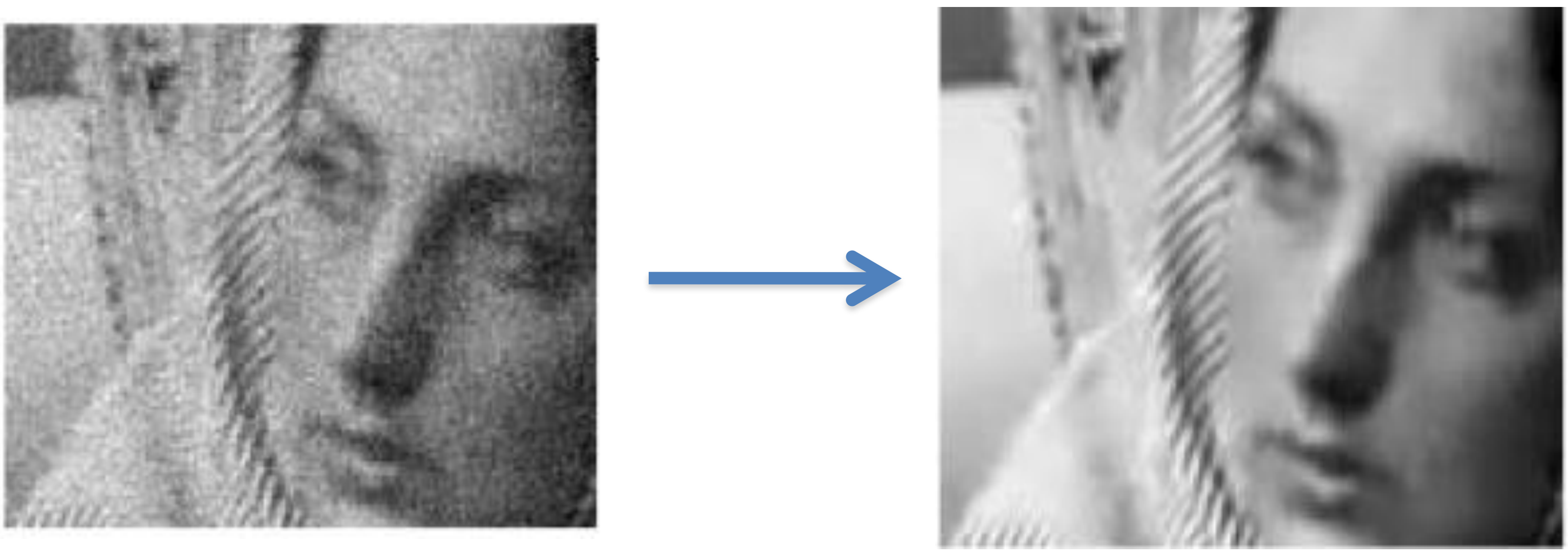


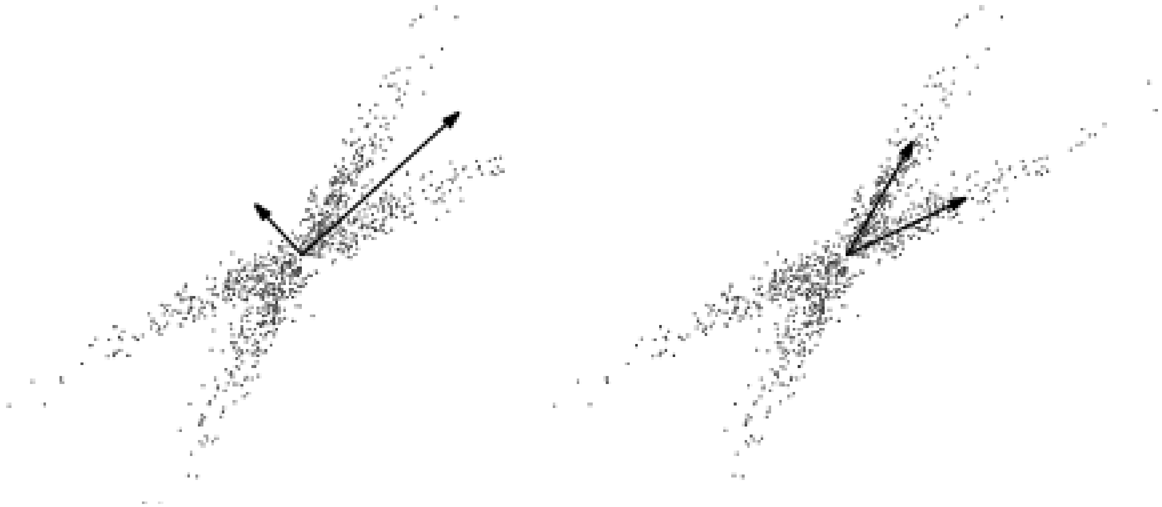
Image Denoising

- Here's what we want



Comparing to Other Techniques

Non-Gaussian data



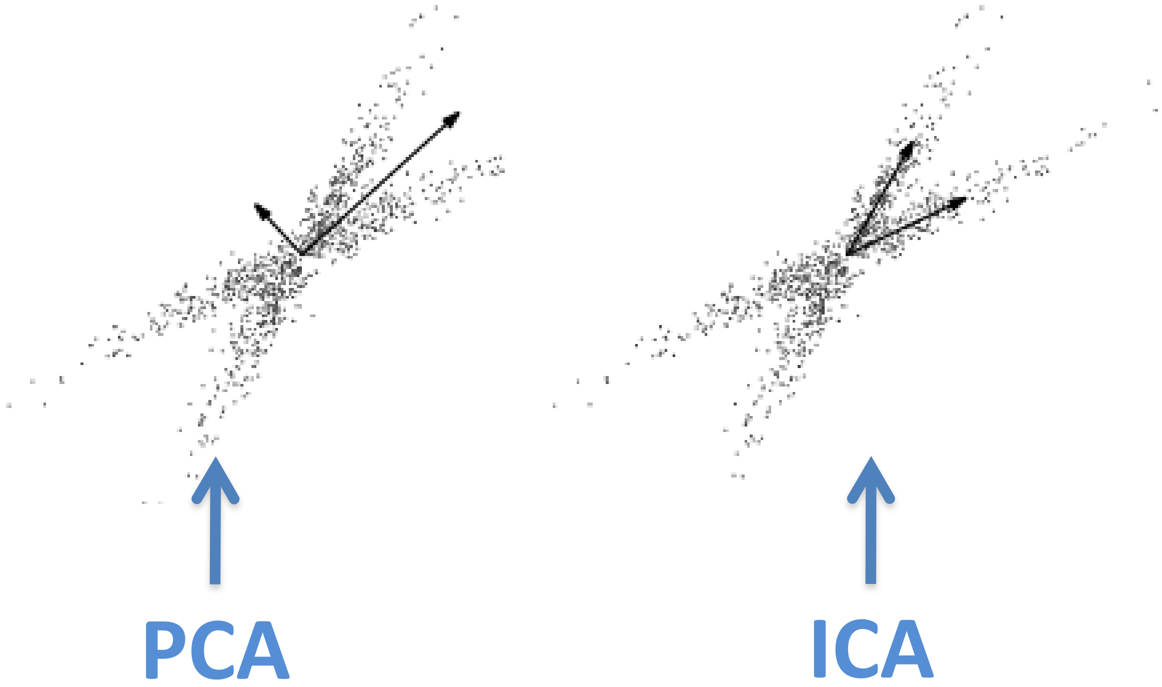
PCA of ICA

Which is which?

Images from Lewicki and Sejnowski, *Learning Overcomplete Representations*, 2000.

Comparing to Other Techniques

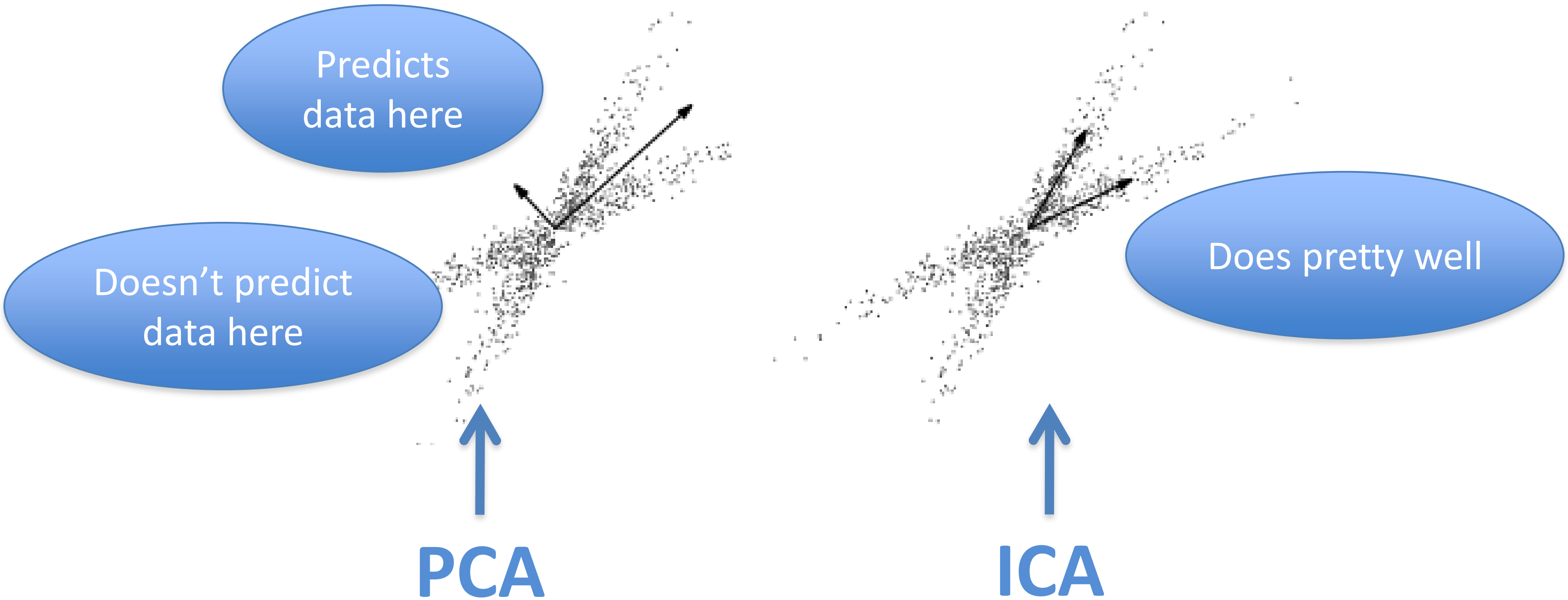
Non-Gaussian data



Images from Lewicki and Sejnowski, *Learning Overcomplete Representations*, 2000.

Comparing to Other Techniques

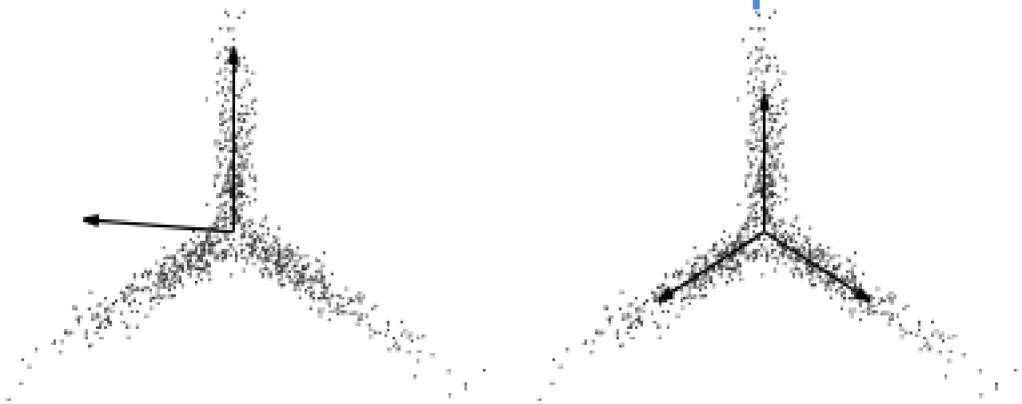
Non-Gaussian data



Images from Lewicki and Sejnowski, *Learning Overcomplete Representations*, 2000.

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.