Sparse Coding and Dictionary Learning

Shihua Zhang

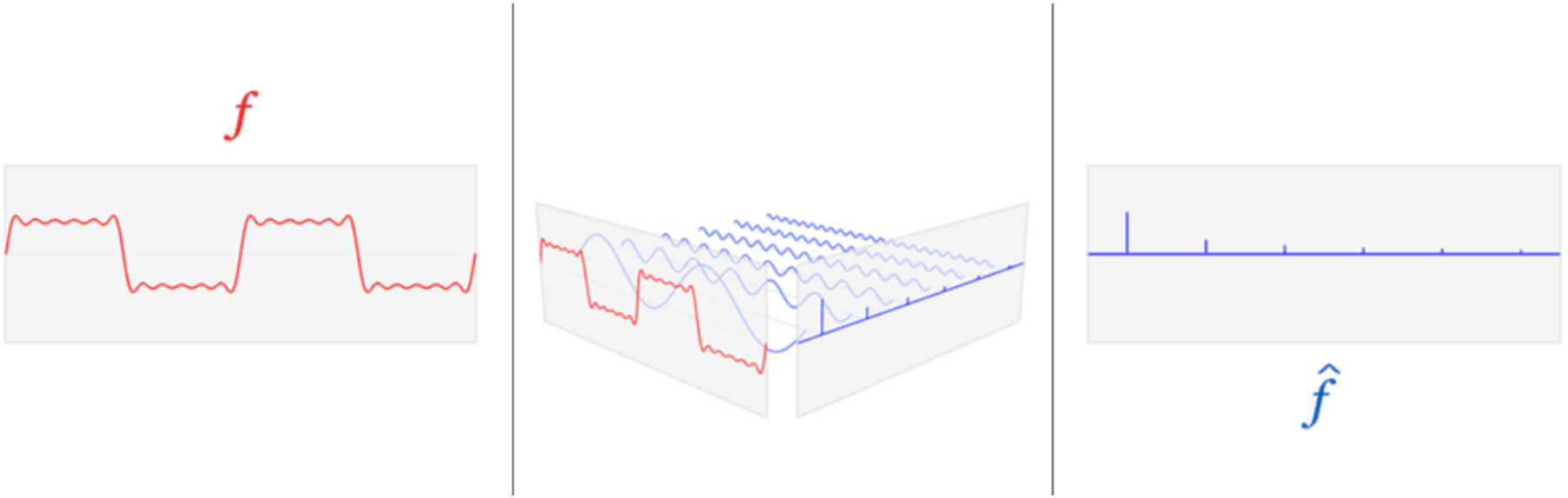
Fall 2019

Overview

Outline

A Brief Review of Signal Transform

- Fourier Transform (1822): breaks a function (or signal) into an alternate representation, characterized by sine and cosines
- A complex function $\Rightarrow$ Linear representation of simple functions



A Brief Review of Signal Transform

- The transform of signal gained tremendous popularity with the birth of the Fast Fourier Transform (FFT) in 1965.
- Other transform method:
  - Discrete Cosine Transform, DCT (1974)
  - Wavelet Transform, WT (1981) and so on

A Brief Review of Signal Transform

- The transform of signal gained tremendous popularity with the birth of the Fast Fourier Transform (FFT) in 1965.
- Other transform method:
  - Discrete Cosine Transform, DCT (1974)
  - Wavelet Transform, WT (1981) and so on
- **Transform** + **Sparse:** method of signal approximation

Figure: JPEG (1992): Sparse + DCT

A Brief Review of Signal Transform

- In 1990, most effective transforms were laid out.
- In 1993, to describe wide scope of patterns in signals, Mallat proposed **a transform scheme** based on selecting small percentage of basis from an overcomplete dictionary.

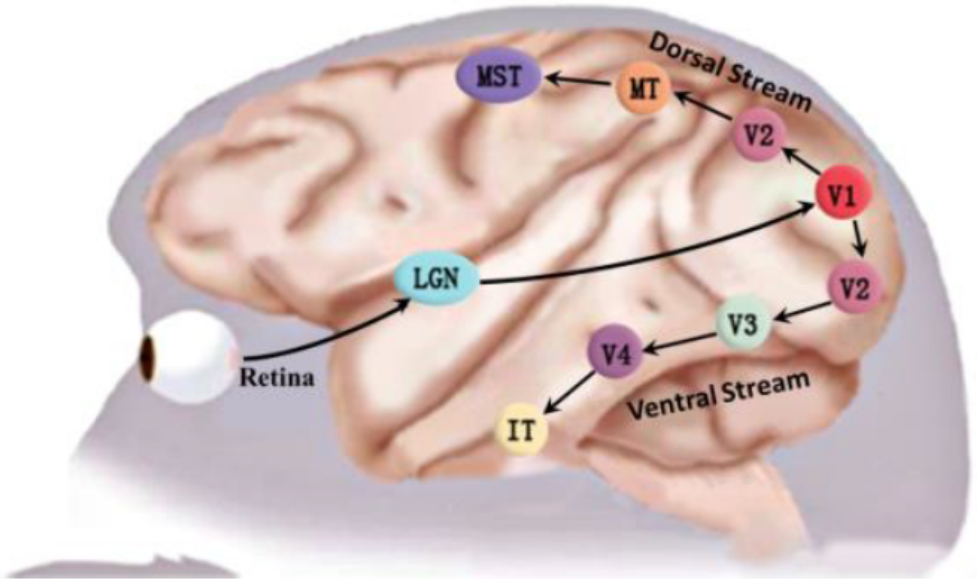
A Brief Review of Signal Transform

- In 1990, most effective transforms were laid out.
- In 1993, to describe wide scope of patterns in signals, Mallat proposed **a transform scheme** based on selecting small percentage of basis from an overcomplete dictionary.
- However, **the analytic dictionaries based on mathematical construction** tend to be over-simplistic compared to the complexity of natural phenomena.

A Brief Review of Signal Transform

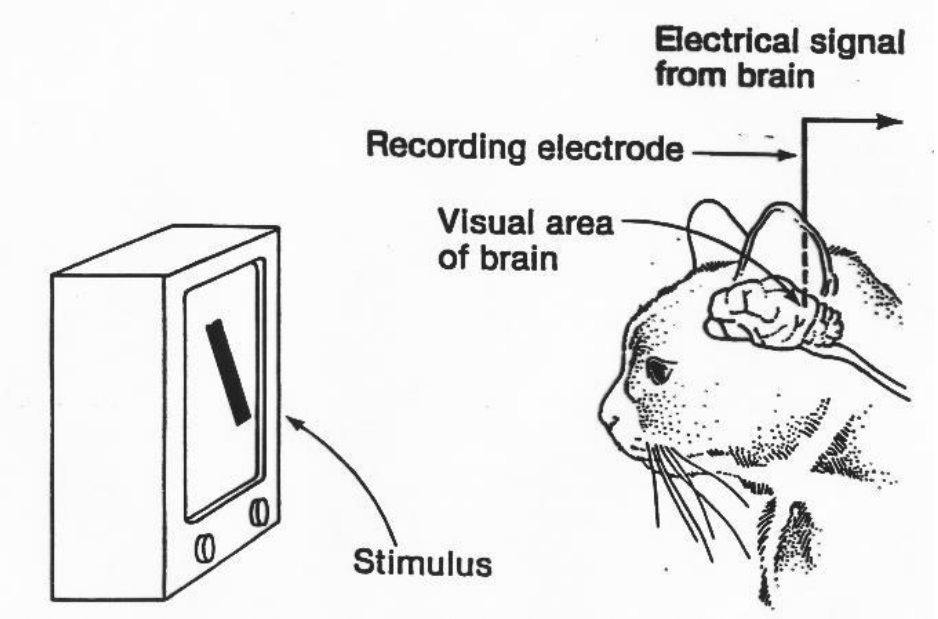
- In 1990, most effective transforms were laid out.
- In 1993, to describe wide scope of patterns in signals, Mallat proposed **a transform scheme** based on selecting small percentage of basis from an overcomplete dictionary.
- However, **the analytic dictionaries based on mathematical construction** tend to be over-simplistic compared to the complexity of natural phenomena.
- Through the 1980's and 1990's, Machine Learning techniques rapidly gained interests, and inspired us to learn from data.
- Almost at the same time, the research on **primary visual cortex** also developed rapidly. **Let's have a look!**

How do Brains Code the Image Signal?



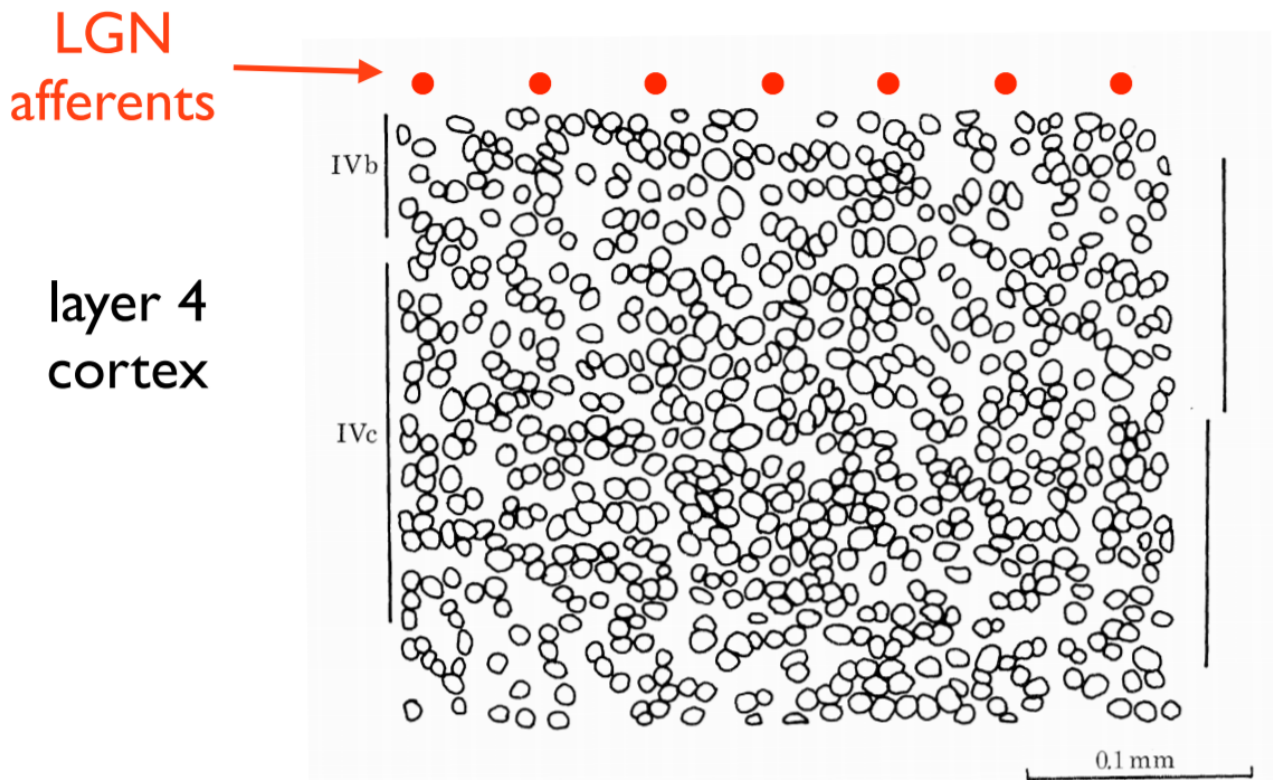
- **Lateral Geniculate Nucleus (LGN)** is a relay center in the thalamus for the visual pathway. It receives a major sensory input from the retina.
- **The primary visual cortex (V1)** is the most studied visual area in the brain. It is the simplest, earliest cortical visual area.

How do Brains Code the Image Signal?



- A simple cell type in the primary visual cortex responds primarily to oriented edges and gratings (bars of particular orientations).
- These cells were discovered by Torsten Wiesel and David Hubel in the late 1950s [Hubel and Wiesel, 1959].

How do Brains Code the Image Signal?



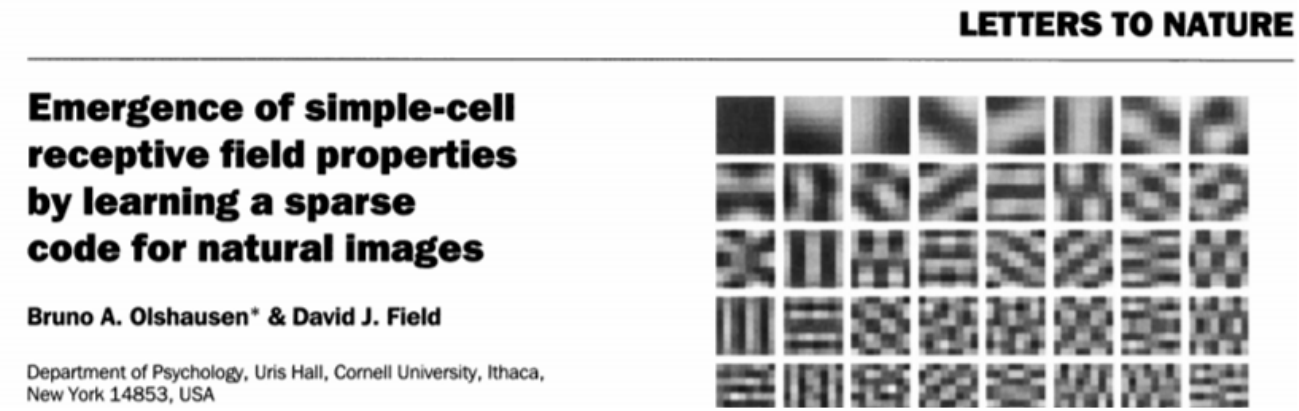
- V1 is **highly overcomplete** (more neurons than input fibers), by factor of at least 100 [Barlow, 1981].
- Cortical neurons have an unusually limited range of firing rates [Barlow, 1981].

The Birth of Sparse Coding

Inspired by signal transform and V1 studies, **sparse coding** of natural images was firstly developed in 1996 [Olshausen and Field, 1996].

Bruno Olshausen David Field

Department of Psychology at Cornell University



LETTERS TO NATURE

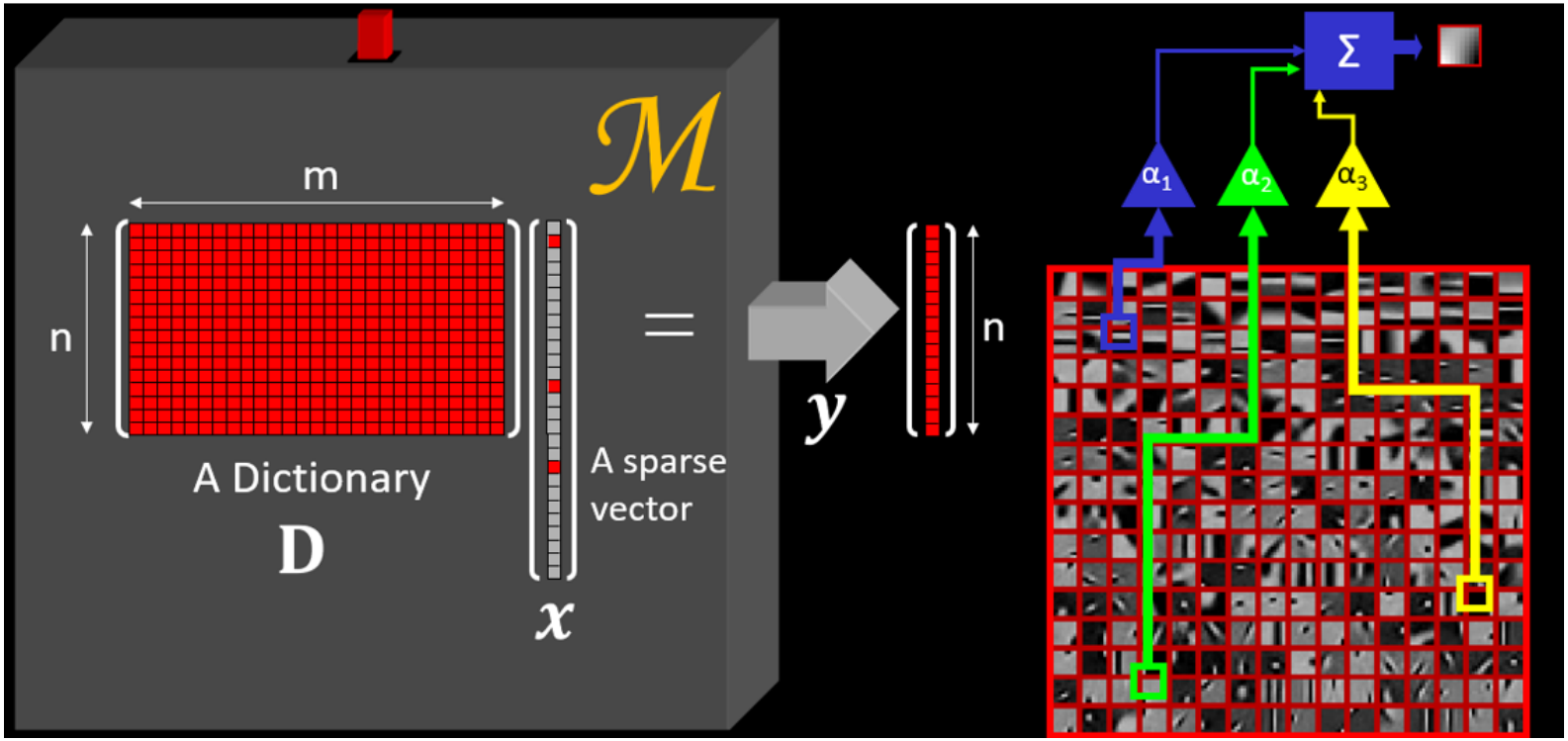
Emergence of simple-cell receptive field properties by learning a sparse code for natural images

Bruno A. Olshausen* & David J. Field

Department of Psychology, Uris Hall, Cornell University, Ithaca, New York 14853, USA

The Model of Sparse Coding

- **Task:** model image patches of size 8×8 pixels
- **Assume** every patch can be described as a linear combination of a few atoms, where the atoms are learned from data



The Power of Sparse Coding

Figure: Image Denoising [Elad and Aharon, 2006]

The Power of Sparse Coding

Figure: Image Inpainting [Xu and Sun, 2010]

The Power of Sparse Coding

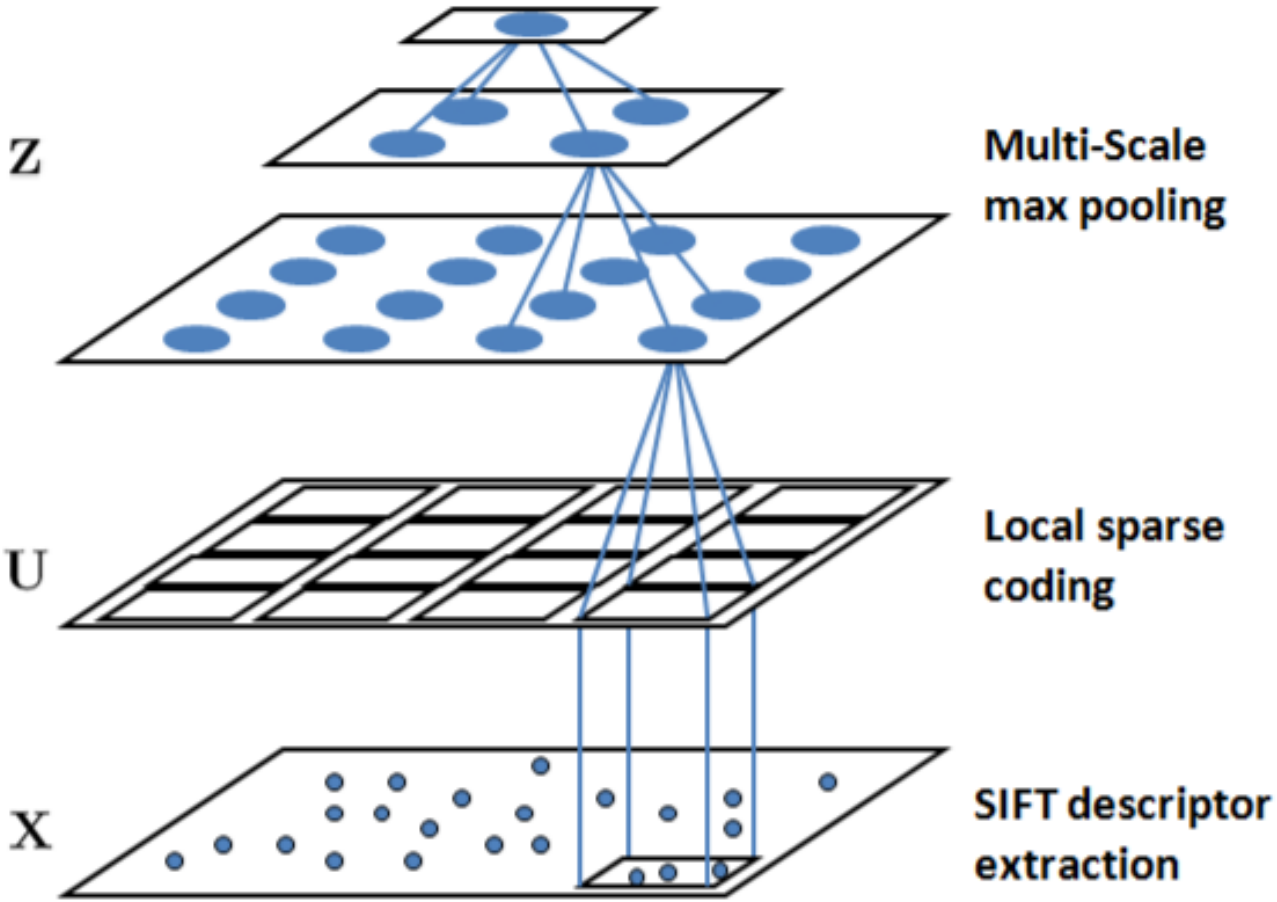


Figure: Image Classification [Yang et al., 2009]

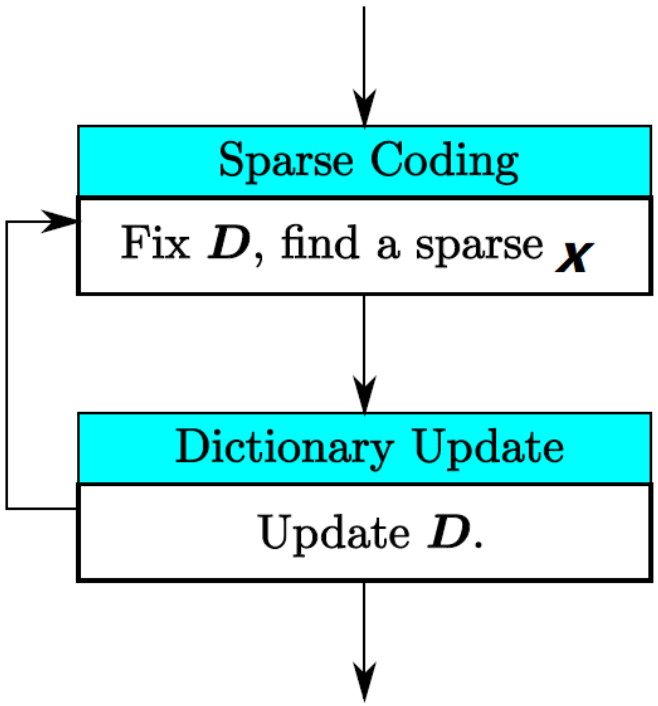
The Framework of Sparse Coding

Consider $\boldsymbol{y} \in \mathbb{R}^{n \times 1}$ is an input signal, $D \in \mathbb{R}^{n \times m}$ is an overcomplete dictionary, where $m \gg n$, $\boldsymbol{x} \in \mathbb{R}^{m \times 1}$ is a sparse representation, $P(\cdot)$ is a regularization term to ensure sparseness.

The Framework of Sparse Coding

Consider $\boldsymbol{y} \in \mathbb{R}^{n\times 1}$ is an input signal, $D \in \mathbb{R}^{n\times m}$ is an overcomplete dictionary, where $m \gg n$, $\boldsymbol{x} \in \mathbb{R}^{m\times 1}$ is a sparse representation, $P(\cdot)$ is a regularization term to ensure sparseness.
Let's rewrite the problem as follows:

$$\min_{\boldsymbol{x},D} \|\boldsymbol{y} - D\boldsymbol{x}\|_2^2 + \lambda P(\boldsymbol{x})$$



Difficulties of Sparse Coding Model

- **Problem 1:** Given a signal and a dictionary, how can we find its atom decomposition?

Difficulties of Sparse Coding Model

- **Problem 1:** Given a signal and a dictionary, how can we find its atom decomposition?
- A simple example:
  - There are 2000 atoms in the dictionary
  - The signal is known to be built of 15 atoms. Then it turns to have $\begin{pmatrix} 2000 \\ 15 \end{pmatrix} \approx 2.4 \times 10^{37}$ possibilities.
  - If each of these takes 1 nano-sec to test, this will take about 7.5×10^{20} years to finish!

Difficulties of Sparse Coding Model

- **Problem 1:** Given a signal and a dictionary, how can we find its atom decomposition?
- A simple example:
  - There are 2000 atoms in the dictionary
  - The signal is known to be built of 15 atoms. Then it turns to have $\begin{pmatrix} 2000 \\ 15 \end{pmatrix} \approx 2.4 \times 10^{37}$ possibilities.
  - If each of these takes 1 nano-sec to test, this will take about 7.5×10^{20} years to finish!

- Efficient algorithms will be introduced in the following:
  - Greedy Algorithms
  - Relaxation Algorithms

Difficulties of Sparse Coding Model

- **Problem 2:** Given a family of signals, how do we find the dictionary to represent it well?

Difficulties of Sparse Coding Model

- **Problem 2:** Given a family of signals, how do we find the dictionary to represent it well?

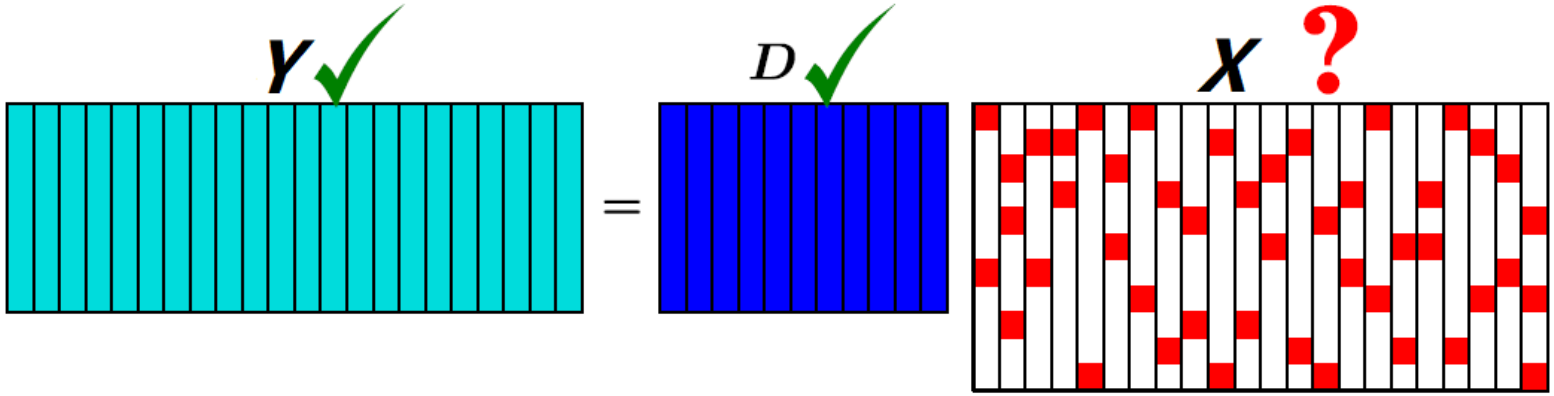
- **Solution:** Learn!
- Efficient algorithms will be introduced in the following:
  - MOD
  - K-SVD
  - Structured Dictionary

Summary

- **Signal Transform:**
  - Signal transform method breaks a complex signal into a sum of simple basis signal.
  - Sparse transform is an efficient method for signal compression.
  - Overcomplete dictionary was put forward to describe wide scope of patterns in signal.
- **Primary Visual Cortex:**
  - V1 is highly overcomplete $\leftrightarrow$ Overcomplete dictionary
  - Limited range of firing rates $\leftrightarrow$ Sparse coding
  - Respond to oriented edges $\leftrightarrow$ Oriented filters
- **Sparse coding** can handle various tasks, including image processing, audio processing, bioinformatics and so on.
- The main framework of sparse coding includes two steps:
  - Sparse Coding
  - Dictionary Learning

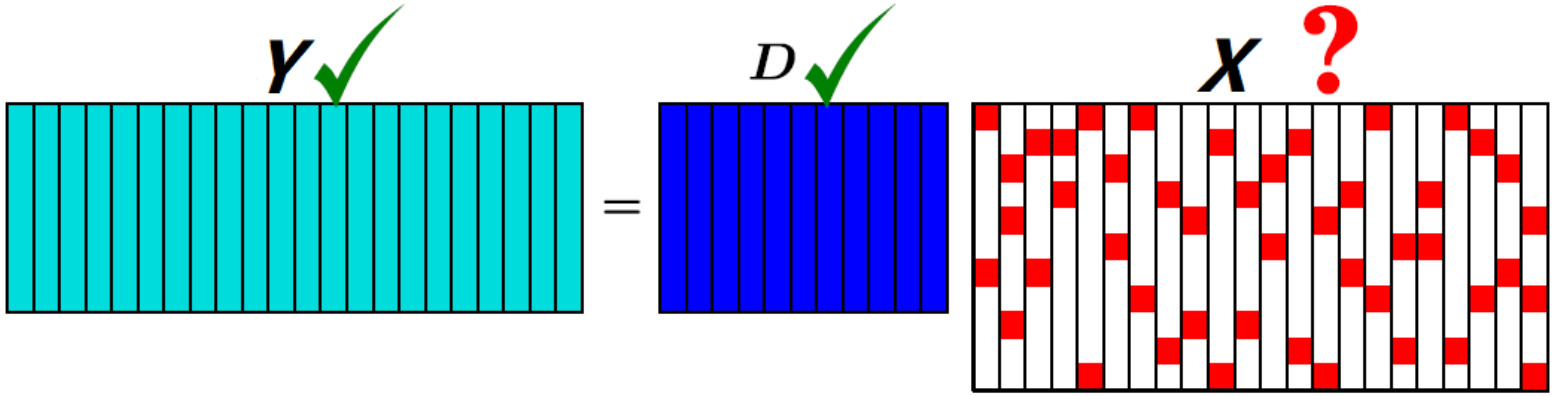
Outline

Algorithms of Sparse Coding



$$\min \|X\|_0 \quad \text{s.t.} \|Y - DX\|_F^2 \leqslant \epsilon$$

Algorithms of Sparse Coding



$$\min \|X\|_0 \quad \text{s.t.} \|Y - DX\|_F^2 \leqslant \epsilon$$

This is an NP-hard problem! [Natarajan, 1995]

- Greedy Algorithms
- Convex Relaxation

Complexity of l_0 Minimization

- **Task:** provide a proof sketch
- **Main idea:** formulate the Minimum Relevant Variables (MRV) problem (which is essentially the l_0-norm minimization) and the 3-set cover problem (X3C), and **show the direct reduction of X3C to MRV**.

Complexity of l_0 Minimization

- **Task:** provide a proof sketch
- **Main idea:** formulate the Minimum Relevant Variables (MRV) problem (which is essentially the l_0-norm minimization) and the 3-set cover problem (X3C), and **show the direct reduction of X3C to MRV**.

Minimum Relevant Variables (MRV):

- **Instance:** A matrix A of size $n \times m$ with integer coefficients and a vector $\boldsymbol{b}$ of size n with integer coefficients.
- **Question:** What is the minimal number of nonzeros that an n-dimensional vector $\boldsymbol{x}$ with rational coefficients can have, given that $\boldsymbol{x}$ is the solution of $A\boldsymbol{x} = \boldsymbol{b}$?

Complexity of l_0 Minimization

Exact Cover by 3 Sets:

- **Instance:** A finite set X with $|X| = 3q$ and a collection C of 3-element subsets of X.
- **Question:** Does C contain an exact cover for X, or a sub-collection $C' \subseteq C$ such that every element of X occurs in exactly one element of C'?

Complexity of l_0 Minimization

Exact Cover by 3 Sets:

- **Instance:** A finite set X with $|X| = 3q$ and a collection C of 3-element subsets of X.
- **Question:** Does C contain an exact cover for X, or a sub-collection $C' \subseteq C$ such that every element of X occurs in exactly one element of C'?

Example:

- Suppose X was $\{1, 2, 3, 4, 5, 6\}$.
- If C was $\{\{1, 2, 3\}, \{2, 3, 4\}, \{1, 2, 5\}, \{2, 5, 6\}, \{1, 5, 6\}\}$ then we could choose C' to be $\{\{2, 3, 4\}, \{1, 5, 6\}\}$.
- If instead, C was $\{\{1, 2, 3\}, \{2, 4, 5\}, \{2, 5, 6\}\}$, then any C' we choose will not be an exact cover.

Exact Cover by 3 Sets (X3C) is one of the basic problems in the set of NP-complete problems [Johnson and Garey, 1979].

Complexity of l_0 Minimization

Let's show the direct reduction of X3C to MRV!

- The set X with size $|X| = 3q$ we map into $\mathbb{R}^{3q}$, subsets C_i from C we map into matrix $A = (a_{j,i})$ of size $3q \times |C|$, where $a_{j,i} = 1$ if and only if C_i covers point corresponding to the j-th coordinate of $\mathbb{R}^{3q}$, and 0 otherwise.
- Vector $\boldsymbol{x}$ is of size $|C|$ and corresponds to the choosen subsets of C ($x_j = 1$ if and only if the covering subset S_j is choosen, and 0 otherwise).
- As a vector $\boldsymbol{b}$ we choose the vector with all $3q$ coefficients equal to 1. Assuming this mapping, the solution of the X3C becomes the minimal solution of MRV, since the number of nonzero coefficients should be at least q.

Greedy Algorithms

Matching Pursuit Algorithms [Mallat and Zhang, 1993]

- **Main idea:** Iteratively choose the best atom from the dictionary to approximately obtain the sparse solution.

Algorithm 1 Matching Pursuit Algorithms

1: Input: Signal $\boldsymbol{y}$, dictionary $D = [\boldsymbol{d}_1, ..., \boldsymbol{d}_m]$ with normalized columns.
2: Initialization: $\boldsymbol{r}^{(0)} = \boldsymbol{y}$, $i = 1$.
3: **while** $i < L$ and $\|\boldsymbol{r}^{(i)}\| < \epsilon$ **do**
4: Find $\boldsymbol{d}_{k_i}$ in D with maximum inner product $|\langle \boldsymbol{r}^{(i)}, \boldsymbol{d}_{k_i} \rangle|$;
5: $a^{(i)} = \langle r^{(i)}, \boldsymbol{d}_{k_i} \rangle$;
6: $\boldsymbol{r}^{(i+1)} = \boldsymbol{r}^{(i)} - a^{(i)} \boldsymbol{d}_{k_i}$;
7: $i = i + 1$;
8: **end while**

Greedy Algorithms

A brief analysis of Matching Pursuit

- As at each step, the residual is orthogonal to the selected filter:

$$\boldsymbol{r}^{(i)} = \left\langle \boldsymbol{r}^{(i)}, \boldsymbol{d}_{k_i} \right\rangle \boldsymbol{d}_{k_i} + \boldsymbol{r}^{(i+1)} \Rightarrow \left\| \boldsymbol{r}^{(i)} \right\|_2^2 = \left| \left\langle \boldsymbol{r}^{(i)}, \boldsymbol{d}_{k_i} \right\rangle \right|^2 + \left\| \boldsymbol{r}^{(i+1)} \right\|_2^2$$

- The error $\|\boldsymbol{r}^{(i)}\|_2^2$ decreases monotonically.
- Although asymptotic convergence is guaranteed, the resulting approximation will in general be sub-optimal.

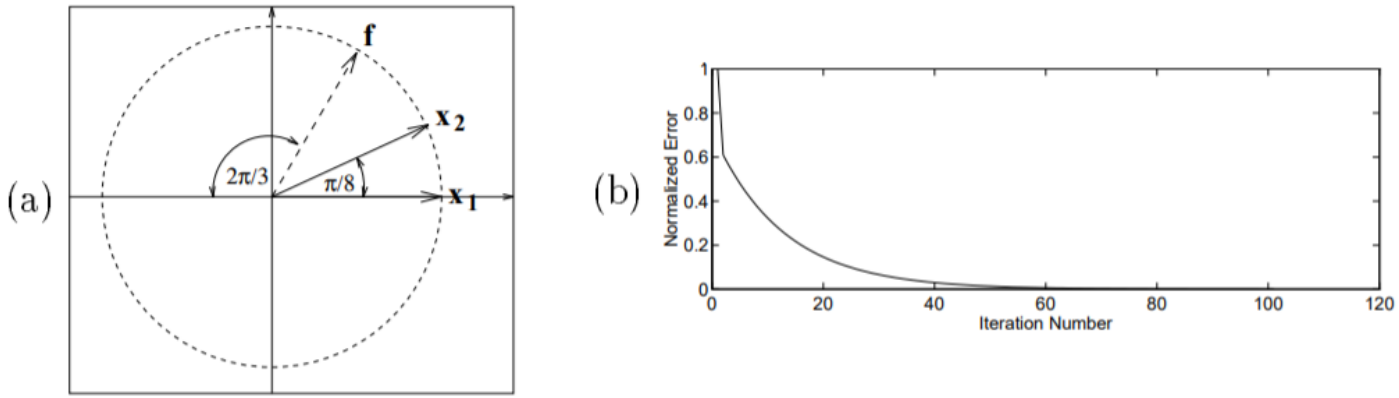


Figure: Matching pursuit example in $\mathbb{R}^2$

Greedy Algorithms

Orthogonal Matching Pursuit [Pati et al., 1993]

- **Main idea:** Maintain full backward orthogonality of the residual at every step and thereby leads to improved convergence.

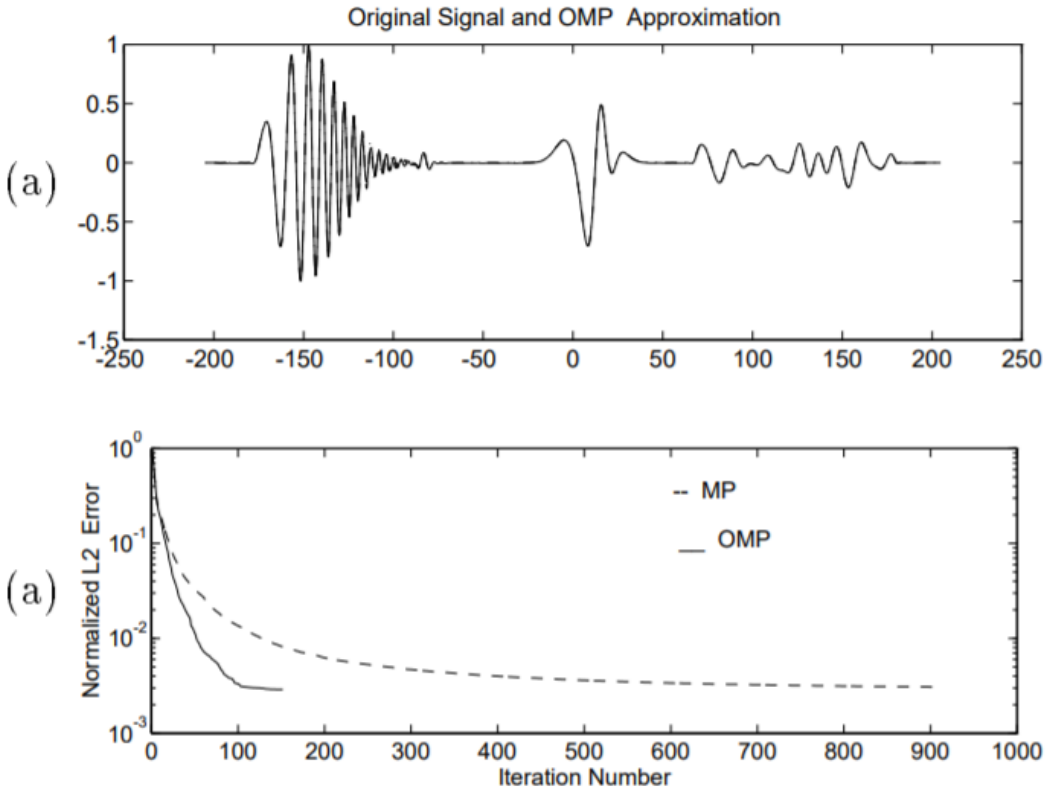
Algorithm 2 Orthogonal Matching Pursuit Algorithms

1: Input: Signal $\boldsymbol{y}$, normalized dictionary $D = [\boldsymbol{d}_1, ..., \boldsymbol{d}_m]$.
2: Initialization: $t = 1$, $\boldsymbol{r}^{(0)} = \boldsymbol{y}$, $\boldsymbol{x} = \boldsymbol{0}$, $D^{(0)} = \phi$, index set $\Lambda_0 = \phi$;
3: **while** $\left\|\boldsymbol{r}^{(t)}\right\| > \tau$ and $t < T$ **do**
4: Find the best matching, $\lambda_t = \arg\max_{j \notin \Lambda^{(t-1)}} \left|\left\langle \boldsymbol{r}^{(t-1)}, \boldsymbol{d}_j \right\rangle\right|$;
5: Update the index set $\Lambda^{(t)} = \Lambda^{(t-1)} \cup \lambda^{(t)}$ and reconstruct data set $D^{(t)} = \left[D^{(t-1)}, \boldsymbol{d}_{\lambda^{(t)}}\right]$;
6: Compute the projection $\tilde{\boldsymbol{x}}^{(t)} = \arg\min \left\|\boldsymbol{y} - D^{(t)}\tilde{\boldsymbol{x}}\right\|_2^2$;
7: Update the representation residual: $\boldsymbol{r}^{(t)} = \boldsymbol{y} - D^{(t)}\tilde{\boldsymbol{x}}^{(t}$
8: $t = t + 1$;
9: **end while**

Greedy Algorithms

MP vs OMP

- For non-orthogonal dictionaries, OMP will in general converge faster than MP.



Relaxation Algorithms

$$\min \|X\|_0 \quad \text{s.t.} \|Y - DX\|_F^2 \leqslant \epsilon$$

- This is an NP-hard problem.
- Greedy algorithms (MP and OMP) provide an approximate solution.
- Another way is to consider the **convex relaxation**.

Relaxation Algorithms

We relax the l_0 and replace it with an l_1:

$$\min \|X\|_1 \quad \text{s.t. } \|Y - DX\|_F^2 \leqslant \epsilon$$

- The solution obtained via l_1-norm minimization constraint is content with the **condition of sparsity**.
- The solution using l_1-norm minimization with sufficient sparsity can be equivalent to the solution obtained by l_0-norm minimization with full probability [Donoho, 2006].
- Moreover, the l_1-norm optimization problem can be solved in polynomial time [Donoho, 2006].

Relaxation Algorithms

A Geometric Interpretation: l_0 to l_1

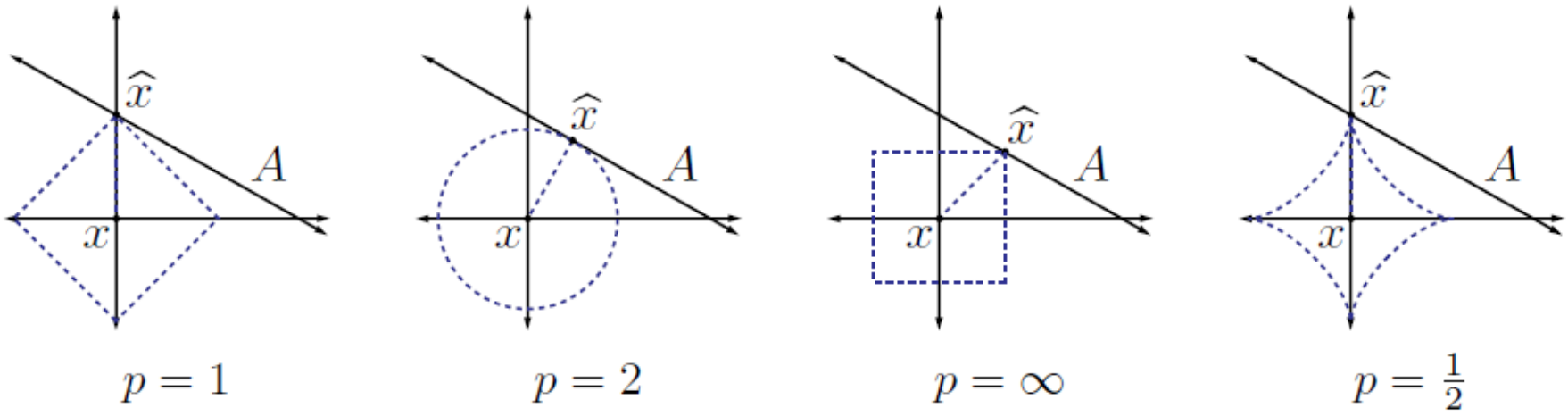


Figure: Best approximation of a point in $\mathbb{R}^2$ by a one-dimensional subspace using the l_p norms

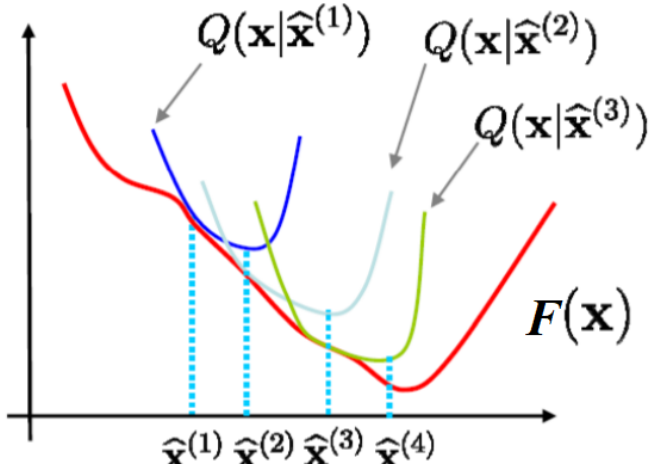
An Overview of Relaxation Algorithms

- **Iterative Shrinkage/Thresholding Algorithm** [Daubechies et al., 2004]
- l_1-Regularized Least Squares [Koh et al., 2007]
- Gradient Projection Sparse Reconstruction [Figueiredo et al., 2007]
- **Coordinate Descent** [Li and Osher, 2009]
- **Fast Iterative Shrinkage/Thresholding Algorithm** [Beck and Teboulle, 2009]
- Alternating Direction Method [Boyd et al., 2011]
- Dual Augmented Lagrangian Method [Yang et al., 2013]
- ...

The Majorization-Minimization (MM) Idea

- **Aim:** minimize $F(\boldsymbol{x})$ – Suppose it is too hard
- Define a function $Q(\boldsymbol{x}, \boldsymbol{x}^{(k)})$ that satisfies [Bertsekas, 1999]:
  - $Q(\boldsymbol{x}^{(k)}, \boldsymbol{x}^{(k)}) = F(\boldsymbol{x}^{(k)})$
  - $Q(\boldsymbol{x}, \boldsymbol{x}^{(k)}) \geqslant F(\boldsymbol{x})$ for all $\boldsymbol{x}$
  - The minimization of $Q(\boldsymbol{x}, \boldsymbol{x}^{(k)})$ is easy to solve.
- Then, the following algorithm necessarily converges to a local (global if $F(\boldsymbol{x})$ is convex) minimization of $F(\boldsymbol{x})$

$$\overset{\boldsymbol{x}^{(0)}}{\Longrightarrow} \min Q(\boldsymbol{x}, \boldsymbol{x}^{(0)}) \overset{\boldsymbol{x}^{(1)}}{\Longrightarrow} \min Q(\boldsymbol{x}, \boldsymbol{x}^{(1)}) \overset{\boldsymbol{x}^{(2)}}{\Longrightarrow} \cdots \overset{\boldsymbol{x}^{(k)}}{\Longrightarrow} \min Q(\boldsymbol{x}, \boldsymbol{x}^{(k)})$$



Proximal Gradient Method [Rockafellar, 1970]

Consider a special type of nonsmooth problems:

$$\min_{\boldsymbol{x}} F(\boldsymbol{x}) = \min_{\boldsymbol{x}} \{f(\boldsymbol{x}) + h(\boldsymbol{x})\}$$

- $f(\boldsymbol{x})$ is an L-smooth and convex function, i.e.:

$$f(\boldsymbol{x}) \leqslant f(\boldsymbol{x}_0) + \nabla f(\boldsymbol{x}_0)^T(\boldsymbol{x} - \boldsymbol{x}_0) + \frac{L}{2}\|\boldsymbol{x} - \boldsymbol{x}_0\|_2^2$$

- $h(\boldsymbol{x})$ is a non-smooth and convex function.

Proximal Gradient Method [Rockafellar, 1970]

Consider a special type of nonsmooth problems:

$$\min_{\boldsymbol{x}} F(\boldsymbol{x}) = \min_{\boldsymbol{x}} \{f(\boldsymbol{x}) + h(\boldsymbol{x})\}$$

- $f(\boldsymbol{x})$ is an L-smooth and convex function, i.e.:

$$f(\boldsymbol{x}) \leqslant f(\boldsymbol{x}_0) + \nabla f(\boldsymbol{x}_0)^T(\boldsymbol{x} - \boldsymbol{x}_0) + \frac{L}{2}\|\boldsymbol{x} - \boldsymbol{x}_0\|_2^2$$

- $h(\boldsymbol{x})$ is a non-smooth and convex function.

Question: How to minimize $F(\boldsymbol{x})$ via the MM idea?

Proximal Gradient Method [Rockafellar, 1970]

Consider a special type of nonsmooth problems:

$$\min_{\boldsymbol{x}} F(\boldsymbol{x}) = \min_{\boldsymbol{x}} \{f(\boldsymbol{x}) + h(\boldsymbol{x})\}$$

- $f(\boldsymbol{x})$ is an L-smooth and convex function, i.e.:

$$f(\boldsymbol{x}) \leqslant f(\boldsymbol{x}_0) + \nabla f(\boldsymbol{x}_0)^T(\boldsymbol{x} - \boldsymbol{x}_0) + \frac{L}{2}\|\boldsymbol{x} - \boldsymbol{x}_0\|_2^2$$

- $h(\boldsymbol{x})$ is a non-smooth and convex function.

Question: How to minimize $F(\boldsymbol{x})$ via the MM idea?
Answer: We need to construct $Q(\boldsymbol{x}, \boldsymbol{x}^{(k)})$ wisely!

Proximal Gradient Method

Consider:

$$Q(\boldsymbol{x}, \boldsymbol{x}^{(k)}) = f(\boldsymbol{x}^{(k)}) + \nabla f(\boldsymbol{x}^{(k)})^T(\boldsymbol{x} - \boldsymbol{x}^{(k)}) + \frac{L}{2}\|\boldsymbol{x} - \boldsymbol{x}^{(k)}\|_2^2 + h(\boldsymbol{x})$$

- $Q(\boldsymbol{x}^{(k)}, \boldsymbol{x}^{(k)}) = F(\boldsymbol{x}^{(k)})$?

Proximal Gradient Method

Consider:

$$Q(\boldsymbol{x}, \boldsymbol{x}^{(k)}) = f(\boldsymbol{x}^{(k)}) + \nabla f(\boldsymbol{x}^{(k)})^T(\boldsymbol{x} - \boldsymbol{x}^{(k)}) + \frac{L}{2}\|\boldsymbol{x} - \boldsymbol{x}^{(k)}\|_2^2 + h(\boldsymbol{x})$$

- $Q(\boldsymbol{x}^{(k)}, \boldsymbol{x}^{(k)}) = F(\boldsymbol{x}^{(k)})$?

  Definitely!
- $Q(\boldsymbol{x}, \boldsymbol{x}^{(k)}) \geqslant F(\boldsymbol{x})$ for all $\boldsymbol{x}$?

Proximal Gradient Method

Consider:

$$Q(\boldsymbol{x}, \boldsymbol{x}^{(k)}) = f(\boldsymbol{x}^{(k)}) + \nabla f(\boldsymbol{x}^{(k)})^T(\boldsymbol{x} - \boldsymbol{x}^{(k)}) + \frac{L}{2}\|\boldsymbol{x} - \boldsymbol{x}^{(k)}\|_2^2 + h(\boldsymbol{x})$$

- $Q(\boldsymbol{x}^{(k)}, \boldsymbol{x}^{(k)}) = F(\boldsymbol{x}^{(k)})$?
  Definitely!
- $Q(\boldsymbol{x}, \boldsymbol{x}^{(k)}) \geqslant F(\boldsymbol{x})$ for all $\boldsymbol{x}$?
  According to the definition of L-smooth!
- The minimization of $Q(\boldsymbol{x}, \boldsymbol{x}^{(k)})$ is easy to solve?

Proximal Gradient Method

Consider:

$$Q(\boldsymbol{x}, \boldsymbol{x}^{(k)}) = f(\boldsymbol{x}^{(k)}) + \nabla f(\boldsymbol{x}^{(k)})^T(\boldsymbol{x} - \boldsymbol{x}^{(k)}) + \frac{L}{2}\|\boldsymbol{x} - \boldsymbol{x}^{(k)}\|_2^2 + h(\boldsymbol{x})$$

- $Q(\boldsymbol{x}^{(k)}, \boldsymbol{x}^{(k)}) = F(\boldsymbol{x}^{(k)})$?
  Definitely!
- $Q(\boldsymbol{x}, \boldsymbol{x}^{(k)}) \geqslant F(\boldsymbol{x})$ for all $\boldsymbol{x}$?
  According to the definition of L-smooth!
- The minimization of $Q(\boldsymbol{x}, \boldsymbol{x}^{(k)})$ is easy to solve?
  It depends!

Then we have:

$$\boldsymbol{x}^{(k+1)} = \text{argmin}_{\boldsymbol{x}} \; Q(\boldsymbol{x}, \boldsymbol{x}^{(k)})$$

$$= \text{argmin}_{\boldsymbol{x}} \; f(\boldsymbol{x}^{(k)}) + \nabla f(\boldsymbol{x}^{(k)})^T(\boldsymbol{x} - \boldsymbol{x}^{(k)}) + \frac{L}{2}\|\boldsymbol{x} - \boldsymbol{x}^{(k)}\|_2^2 + h(\boldsymbol{x})$$

$$= \text{argmin}_{\boldsymbol{x}} \; h(\boldsymbol{x}) + \frac{L}{2}\|\boldsymbol{x} - (\boldsymbol{x}^{(k)} - \frac{1}{L}\nabla f(\boldsymbol{x}^{(k)}))\|_2^2$$

Proximal Gradient Method

Proximal Mapping:
The proximal mapping (or operator) of a convex function h is:

$$\text{prox}_{\frac{1}{L},h}(\boldsymbol{x}) = \underset{\boldsymbol{u}}{\text{argmin}} \left(h(\boldsymbol{u}) + \frac{L}{2}\|\boldsymbol{u} - \boldsymbol{x}\|_2^2 \right)$$

Examples:

- $h(\boldsymbol{x}) = 0 : \text{prox}_{\frac{1}{L},h}(\boldsymbol{x}) = \boldsymbol{x}$
- $h(\boldsymbol{x}) = I_C(\boldsymbol{x})$ (indicator function of C): $\text{prox}_{\frac{1}{L},h}(\boldsymbol{x})$ is projection on C:

$$\text{prox}_{\frac{1}{L},h}(\boldsymbol{x}) = P_C(\boldsymbol{x}) = \underset{\boldsymbol{u}\in C}{\text{argmin}} \|\boldsymbol{u} - \boldsymbol{x}\|_2^2$$

Proximal Gradient Method

Consider a special type of nonsmooth problems:

$$\min_{\boldsymbol{x}} F(\boldsymbol{x}) = \min_{\boldsymbol{x}} \{f(\boldsymbol{x}) + h(\boldsymbol{x})\}$$

- $f(\boldsymbol{x})$ is an L-smooth and convex function.
- $h(\boldsymbol{x})$ is a non-smooth and convex function.

Proximal gradient algorithm:

$$\boldsymbol{x}^{(k+1)} = \text{prox}_{\frac{1}{L},h}\left(\boldsymbol{x}^{(k)} - \frac{1}{L}\nabla f\left(\boldsymbol{x}^{(k)}\right)\right)$$

- Gradient method: $h(\boldsymbol{x}) = 0$
- Gradient projection method: $h(\boldsymbol{x}) = I_C(\boldsymbol{x})$

Iterative Shrinkage/Thresholding Algorithm

Proximal gradient algorithm:

$$\boldsymbol{x}^{(k+1)} = \text{prox}_{\frac{1}{L},h}\left(\boldsymbol{x}^{(k)} - \frac{1}{L}\nabla f\left(\boldsymbol{x}^{(k)}\right)\right)$$

Consider the sparse coding problem:

$$\arg\min_{\boldsymbol{x}} \frac{1}{2}\|\boldsymbol{y} - D\boldsymbol{x}\|_2^2 + \lambda\|\boldsymbol{x}\|_1$$

$\|\cdot\|_2^2$ is smooth and convex, $\|\cdot\|_1$ is non-smooth and convex.

- Which L is appropriate?
- How to minimize $\text{prox}_{\frac{1}{L},h}(\boldsymbol{x})$?

Which L is Appropriate?

Theorem 1

If f is twice differentiable, then there is an equivalent definition of L-smooth: f is L-smooth if there exists a constant $L > 0$ such that $\nabla^2 f(\boldsymbol{x}) \preceq LI$. where I is an identity matrix. In other words, the largest singular value of the Hessian of f is uniformly upper bounded by L everywhere.

According to above theorem, we can let $L > \sigma_D$, where σ_D is the largest singular value of D.

How to Minimize $\text{prox}_{\frac{1}{L},h}(\boldsymbol{x})$?

$$\begin{aligned}\boldsymbol{x}^{(k+1)} &= \text{prox}_{\frac{1}{L},h}\left(\boldsymbol{x}^{(k)} - \frac{1}{L}\nabla f\left(\boldsymbol{x}^{(k)}\right)\right) \\ &= \arg\min_{\boldsymbol{x}} \frac{L}{2}\|\boldsymbol{x} - (\boldsymbol{x}^{(k)} - \frac{1}{L}D^T(D\boldsymbol{x}^{(k)} - 2\boldsymbol{y}))\|_2^2 + \lambda\|\boldsymbol{x}\|_1\end{aligned}$$

Then the problem turns to the following formulation:

$$\min_{\boldsymbol{x}} \frac{1}{2}\|\boldsymbol{x} - \boldsymbol{z}\|_2^2 + \lambda\|\boldsymbol{x}\|_1$$

- These problems have a closed form solution known as soft-thresholding.

$$S_\lambda(z_i) = \begin{cases} 0 & |z_i| \leqslant \lambda \\ z_i - \lambda\,\text{sign}(z_i) & |z_i| > \lambda \end{cases}$$

Iterative Shrinkage/Thresholding Algorithm

$$\arg\min_{\boldsymbol{x}} \frac{1}{2}\|\boldsymbol{y} - D\boldsymbol{x}\|_2^2 + \lambda\|\boldsymbol{x}\|_1$$

Algorithm 3 ISTA

1: Input: Signal $\boldsymbol{y}$, normalized dictionary D.
2: Initialization: $\boldsymbol{x} = \mathbf{0}$, $L > \sigma_D$;
3: **while** above the threshold **do**
4: $\quad \boldsymbol{x}^{(k+1)} = S_{(\lambda/L)}\left(\boldsymbol{x}^{(k)} - \frac{1}{L}D^T\left(D\boldsymbol{x}^{(k)} - \boldsymbol{y}\right)\right)$
5: **end while**

- ISTA has a worst-case complexity result of $O(\frac{1}{k})$ [Beck and Teboulle, 2009].
- Then we will introduce a new ISTA with an improved complexity result of $O(\frac{1}{k^2})$ [Beck and Teboulle, 2009].

Fast Iterative Shrinkage/Thresholding Algorithm

Algorithm 4 FISTA

1: Input: Signal $\boldsymbol{y}$, normalized dictionary D.
2: Initialization: $\boldsymbol{x} = \mathbf{0}$, $L > \sigma_D$;
3: **while** above the threshold **do**
4: $\quad \boldsymbol{x}^{(k+1)} = S_{\frac{\lambda}{L}}\left(\boldsymbol{z}^{(k)} - \frac{1}{L}D^T\left(D\boldsymbol{z}^{(k)} - \boldsymbol{y}\right)\right)$
5: $\quad t^{(k+1)} = \frac{1+\sqrt{1+4t^{(k)^2}}}{2}$
6: $\quad \boldsymbol{z}^{(k+1)} = \boldsymbol{x}^{(k+1)} + \left(\frac{t^{(k)}-1}{t^{(k+1)}}\right)\left(\boldsymbol{x}^{(k+1)} - \boldsymbol{x}^{(k)}\right)$
7: **end while**

- The **main difference** between the above algorithm and ISTA: the iterative-shrinkage step (4) is not employed on the previous point $\boldsymbol{x}^{(k)}$, but rather a very specific linear combination of the previous two points $\{\boldsymbol{x}^{(k-1)}, \boldsymbol{x}^{(k)}\}$.

ISTA vs FISTA

- We generated an instance of the problem with $\lambda = 1$, $D \in \mathbb{R}^{100\times 110}$
- The true vector is $\boldsymbol{d}_3 - \boldsymbol{d}_7$
- We ran 200 iterations of ISTA and FISTA.

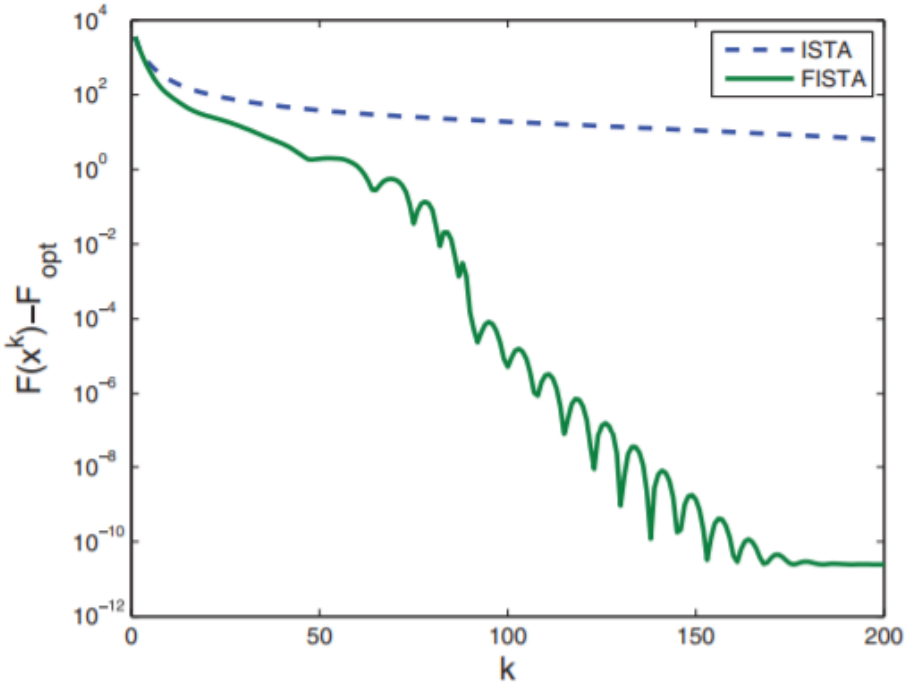


Figure: Results of 200 iterations of ISTA and FISTA

ISTA vs FISTA

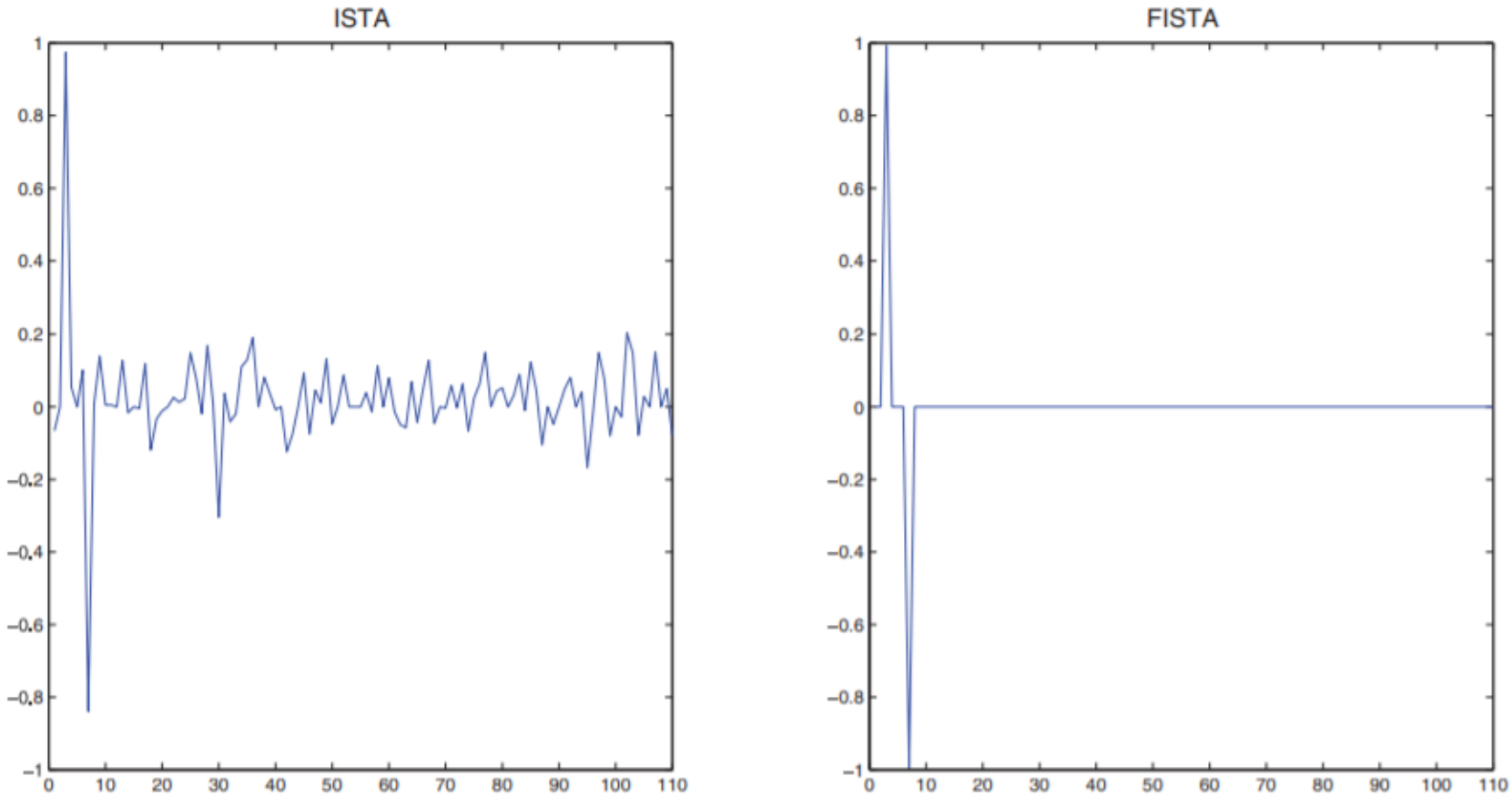
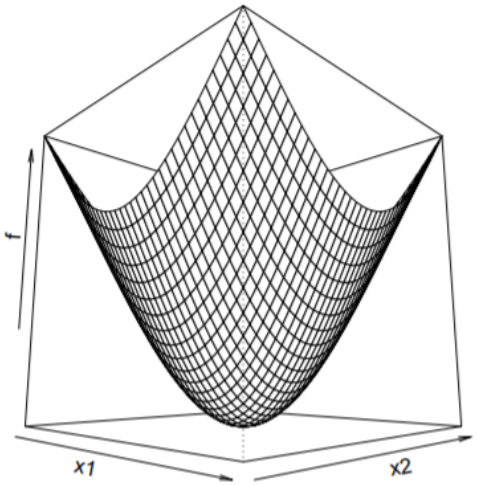


Figure: Solutions obtained by ISTA (left) and FISTA (right).

Coordinate-Wise Minimization

Question: Given convex, differentiable $f : \mathbb{R}^n \to \mathbb{R}$, if we are at a point $\boldsymbol{x}$ such that $f(\boldsymbol{x})$ is minimized along each coordinate axis, have we found a global minimizer? i.e.:
Does $f(\boldsymbol{x} + d\boldsymbol{e}_i) \geqslant f(\boldsymbol{x})$ for all $d, i \Rightarrow f(\boldsymbol{x}) = \min_{\boldsymbol{z}} f(\boldsymbol{z})$?

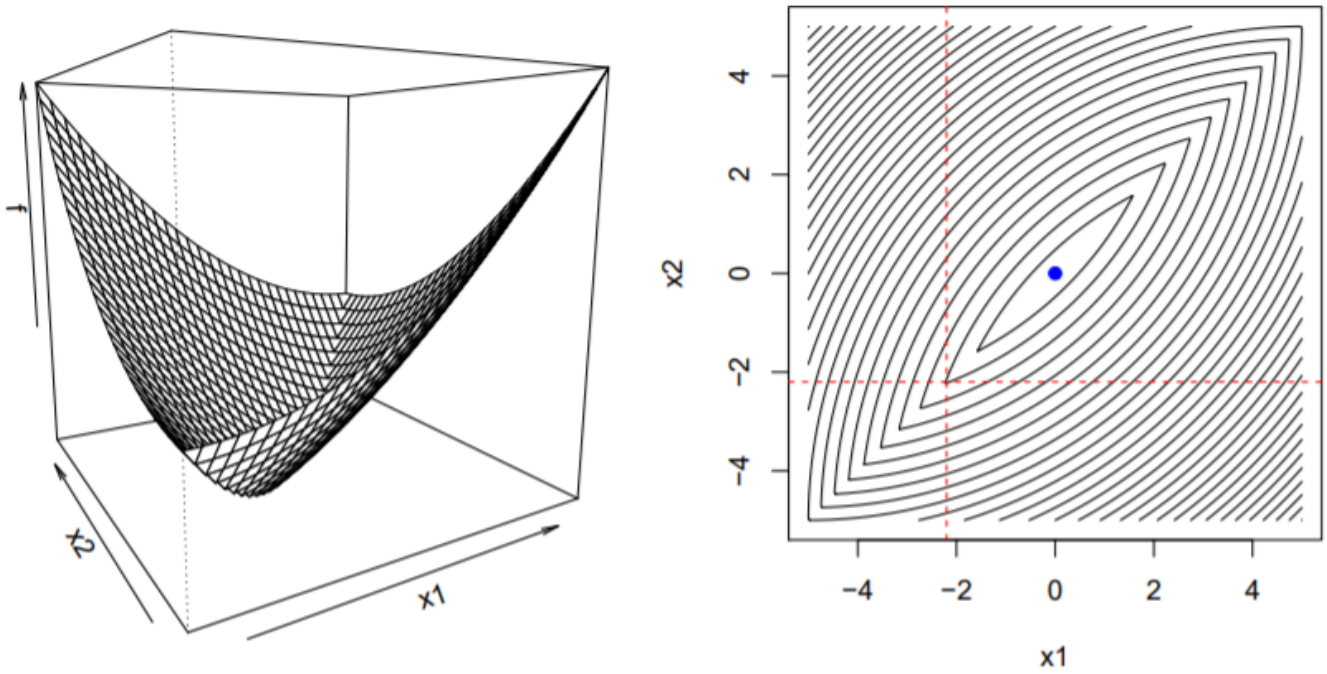


Answer: Yes!

$$\nabla f(\boldsymbol{x}) = \left(\frac{\partial f}{\partial x_1}(\boldsymbol{x}), \ldots \frac{\partial f}{\partial x_n}(\boldsymbol{x}) \right) = 0$$

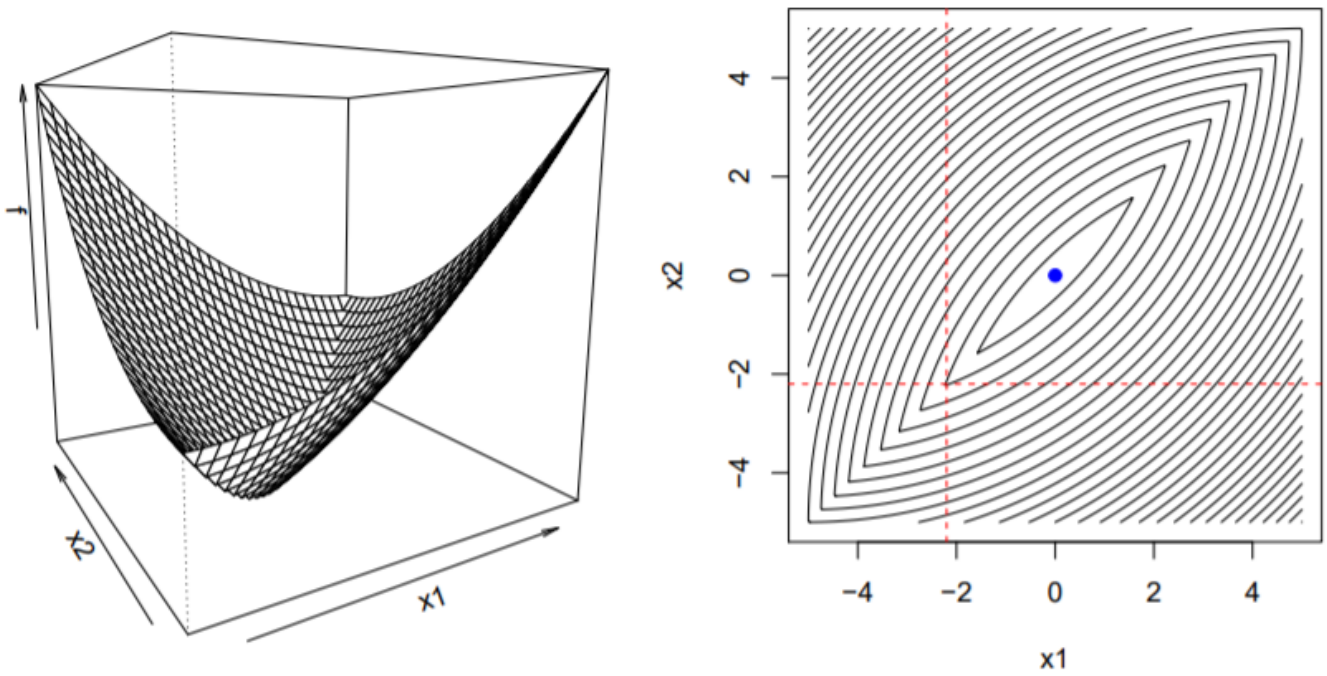
Coordinate-Wise Minimization

Question: Same question, but for f convex (not differentiable) ...?



Coordinate-Wise Minimization

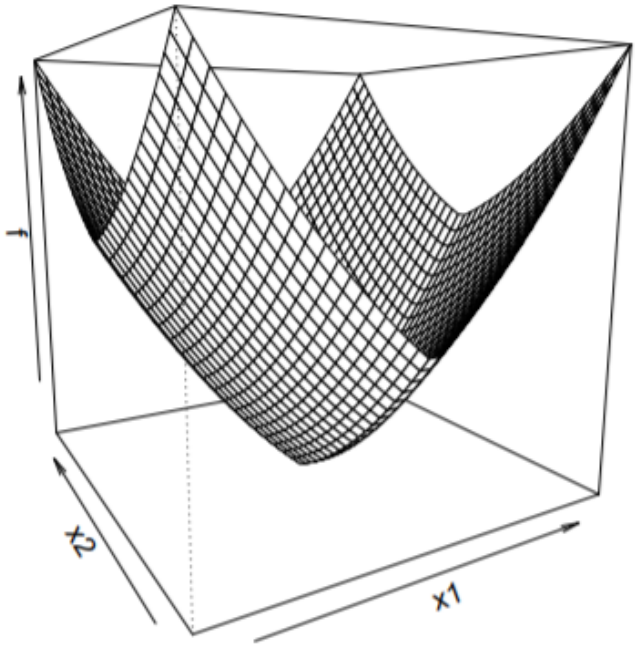
Question: Same question, but for f convex (not differentiable) ...?

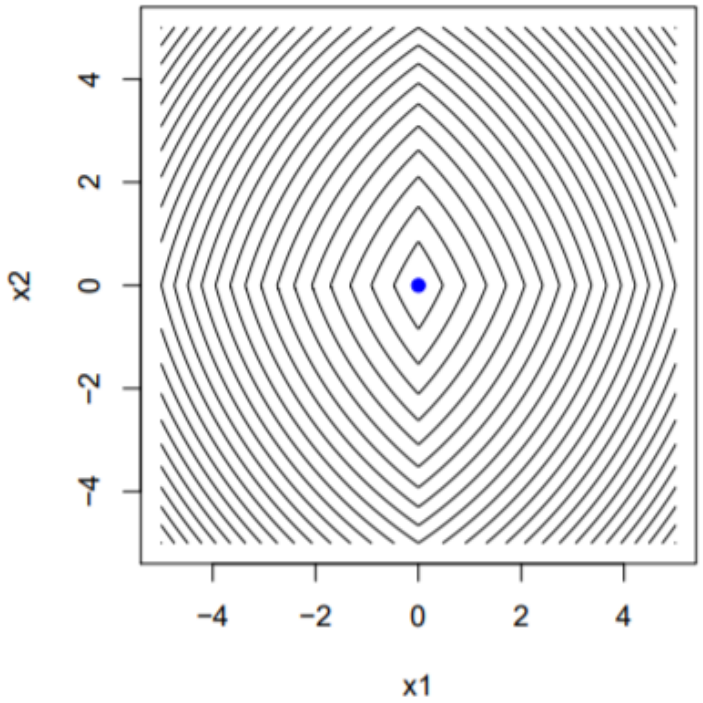


Answer: No! Look at the above counterexample.

Coordinate-Wise Minimization

Question: Same question again, but now $f(\boldsymbol{x}) = g(\boldsymbol{x}) + \sum_{i=1}^{n} h_i(x_i)$, with g convex, differentiable and each h_i convex only ...?





Coordinate-Wise Minimization

Question: Same question again, but now $f(\boldsymbol{x}) = g(\boldsymbol{x}) + \sum_{i=1}^{n} h_i(x_i)$, with g convex, differentiable and each h_i convex only ...?

Coordinate-Wise Minimization

Question: Same question again, but now $f(\boldsymbol{x}) = g(\boldsymbol{x}) + \sum_{i=1}^{n} h_i(x_i)$, with g convex, differentiable and each h_i convex only ...?

Answer: Yes! For any $\boldsymbol{y}$:

$$f(\boldsymbol{y}) - f(\boldsymbol{x}) \geqslant \nabla g(\boldsymbol{x})^T(\boldsymbol{y} - \boldsymbol{x}) + \sum_{i=1}^{n} [h_i(y_i) - h_i(x_i)]$$

$$= \sum_{i=1}^{n} \underbrace{[\nabla_i g(\boldsymbol{x})(y_i - x_i) + h_i(y_i) - h_i(x_i)]}_{\geqslant 0} \geqslant 0$$

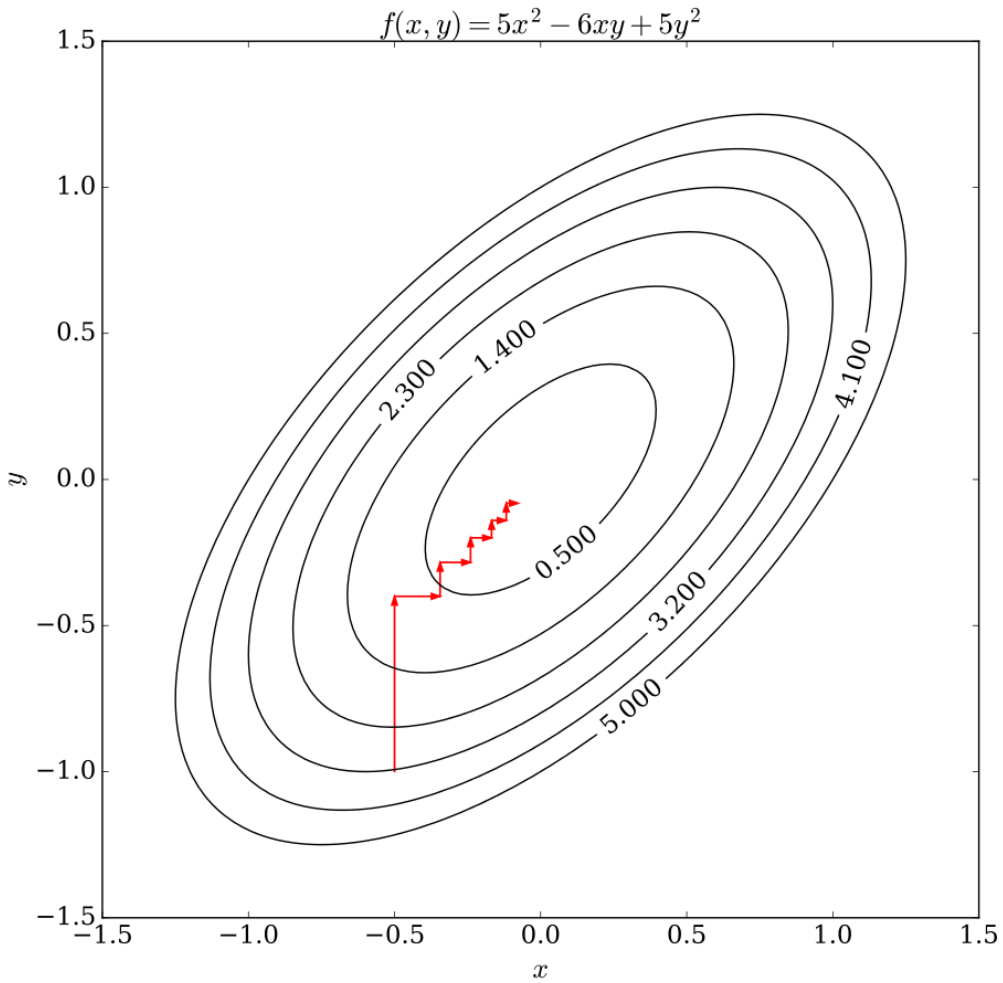
Coordinate Descent [Bertsekas, 1999]

This suggests that for $f(\boldsymbol{x}) = g(\boldsymbol{x}) + \sum_{i=1}^{n} h_i(\boldsymbol{x}_i)$ (with g convex, differentiable and each h_i convex) we can use coordinate descent to find a minimizer: start with some initial guess $\boldsymbol{x}^{(0)}$, and repeat for $k = 1, 2, 3, \ldots$:

$$
\begin{aligned}
x_1^{(k)} &\in \underset{x_1}{\operatorname{argmin}} f\left(x_1, x_2^{(k-1)}, x_3^{(k-1)}, \ldots x_n^{(k-1)}\right) \\
x_2^{(k)} &\in \underset{x_1}{\operatorname{argmin}} f\left(x_1^{(k)}, x_2, x_3^{(k-1)}, \ldots x_n^{(k-1)}\right) \\
x_3^{(k)} &\in \underset{x_3}{\operatorname{argmin}} f\left(x_1^{(k)}, x_2^{(k)}, x_3, \ldots x_n^{(k-1)}\right) \\
&\ldots \\
x_n^{(k)} &\in \underset{x_n}{\operatorname{argmin}} f\left(x_1^{(k)}, x_2^{(k)}, x_3^{(k)}, \ldots x_n\right)
\end{aligned}
$$

Coordinate Descent

The process of coordinate descent:



Sparse Coding via Coordinate Descent

Consider the sparse coding problem:

$$f(\boldsymbol{x}) = \frac{1}{2}\|\boldsymbol{y} - D\boldsymbol{x}\|_2^2 + \lambda\|\boldsymbol{x}\|_1$$

Note that the non-smooth part is separable: $\|\boldsymbol{x}\|_1 = \sum_{i=1}^{n} |x_i|$

Minimizing over x_i, with $x_j, j \neq i$ fixed:

$$\begin{aligned}
&\arg\min_{x_i} \ \frac{1}{2}\|\boldsymbol{y} - D\boldsymbol{x}\|_2^2 + \lambda\|\boldsymbol{x}\|_1 \\
=&\arg\min_{x_i} \ \frac{1}{2}\|(\boldsymbol{y} - D_{-i}\boldsymbol{x}_{-i}) - x_i\boldsymbol{d}_i\|_2^2 + \lambda|x_i| \\
=&S_{\lambda/\|\boldsymbol{d}_i\|^2}\left(\frac{\boldsymbol{d}_i^T(\boldsymbol{y} - D_{-i}\boldsymbol{x}_{-i})}{\boldsymbol{d}_i^T\boldsymbol{d}_i}\right)
\end{aligned}$$

Repeat this for $i = 1, 2, \cdots, n$

Sparse Coding Based Classification Method

Data set ($\#Tr$)	OMP	l_1_ls	PALM	FISTA	DALM	Homotopy	TPTSR
ORL(1)	64.94±2.374	68.50±2.021	68.36±1.957	70.67±2.429	70.22±2.805	66.53±1.264	**71.56±3.032**
ORL(2)	80.59±2.256	**84.84±2.857**	80.66±2.391	84.72±3.242	84.38±2.210	83.88±2.115	83.38±2.019
ORL(3)	89.00±1.291	89.71±1.313	86.82±1.959	90.00±3.141	90.36±1.829	89.32±1.832	**90.71±1.725**
ORL(4)	91.79±1.713	**94.83±1.024**	88.63±2.430	94.13±1.310	94.71±1.289	94.38±1.115	94.58±1.584
ORL(5)	93.75±2.125	**95.90±1.150**	92.05±1.039	95.60±1.761	95.50±1.269	95.60±1.430	95.75±1.439
ORL(6)	95.69±1.120	97.25±1.222	92.06±1.319	96.69±1.319	96.56±1.724	**97.31±1.143**	95.81±1.642
Average Time(5)	0.0038s	0.1363s	7.4448s	0.9046s	0.8013s	0.0100s	**0.0017s**
LFW(3)	22.22±1.369	**28.24±0.667**	15.16±1.202	26.55±0.767	25.46±0.705	26.12±0.831	27.32±1.095
LFW(5)	27.83±1.011	**35.58±1.489**	12.89±1.286	34.13±0.459	33.90±1.181	33.95±1.680	35.43±1.409
LFW(7)	32.76±2.318	40.17±2.061	11.63±0.937	39.86±1.226	38.40±1.890	38.04±1.251	**40.92±1.201**
LFW(9)	35.14±1.136	**44.93±1.123**	7.84±1.278	43.86±1.492	43.56±1.393	42.29±2.721	44.72±1.793
Average Time(7)	**0.0140s**	0.6825s	33.2695s	3.0832s	3.9906s	0.2372s	0.0424s
Extended YaleB(3)	44.20±2.246	63.10±2.341	63.73±2.073	62.84±2.623	**63.76±2.430**	64.22±2.525	56.23±2.153
Extended YaleB(6)	72.48±2.330	81.97±0.850	81.93±0.930	**82.25±0.734**	81.74±1.082	81.64±1.159	78.53±1.731
Extended YaleB(9)	83.42±0.945	88.90±0.544	88.50±1.096	**89.31±0.829**	89.26±0.781	89.12±0.779	86.49±1.165
Extended YaleB(12)	88.23±0.961	**92.49±0.622**	91.07±0.725	92.03±1.248	91.85±0.710	92.03±0.767	91.30±0.741
Extended YaleB(15)	91.97±0.963	94.22±0.719	93.19±0.642	**94.50±0.824**	93.07±0.538	93.67±0.860	93.38±0.785
Average Time(12)	**0.0116s**	3.2652s	17.4516s	1.5739s	1.9384s	0.5495s	0.0198s
COIL20(3)	75.90±1.656	77.62±2.347	70.26±2.646	75.80±2.056	76.67±2.606	**78.46±2.603**	78.16±2.197
COIL20(5)	83.00±1.892	82.63±1.701	79.55±1.153	84.09±2.003	84.38±1.319	**84.58±1.487**	83.69±1.804
COIL20(7)	87.26±1.289	88.22±1.304	82.88±1.445	88.89±1.598	89.00±1.000	**89.36±1.147**	87.75±1.451
COIL20(9)	89.56±1.763	90.97±1.595	84.94±1.563	90.16±1.366	**91.82±1.555**	91.44±1.198	89.41±2.167
COIL20(11)	91.70±0.739	92.98±1.404	87.16±1.184	93.43±1.543	**93.46±1.327**	93.55±1.205	92.71±1.618
COIL20(13)	92.49±1.146	94.29±0.986	88.36±1.283	94.50±0.850	93.92±1.102	**94.93±0.788**	92.72±1.481
Average Time(13)	**0.0038s**	0.0797s	7.5191s	0.7812s	0.7762s	0.0159s	0.0053s
Fifteen scene(3)	85.40±1.388	**86.83±1.082**	86.15±1.504	86.48±1.542	85.89±1.624	86.15±1.073	86.62±1.405
Fifteen scene(6)	89.14±1.033	90.34±0.685	89.97±0.601	90.82±0.921	90.12±0.998	89.65±0.888	**90.83±0.737**
Fifteen scene(9)	83.42±0.945	88.90±0.544	88.50±1.096	89.31±0.829	89.26±0.781	89.12±0.779	**90.64±0.940**
Fifteen scene(12)	91.67±0.970	92.06±0.536	**92.76±0.905**	92.22±0.720	92.45±0.860	92.35±0.706	92.33±0.563
Fifteen scene(15)	93.32±0.609	93.37±0.506	93.63±0.510	93.63±0.787	93.53±0.829	**93.84±0.586**	93.80±0.461
Fifteen scene(18)	93.61±0.334	94.31±0.551	94.67±0.678	94.28±0.396	94.16±0.344	94.16±0.642	**94.78±0.494**
Average Time(18)	**0.0037s**	0.0759s	0.9124s	0.8119s	0.8500s	0.1811s	0.0122s

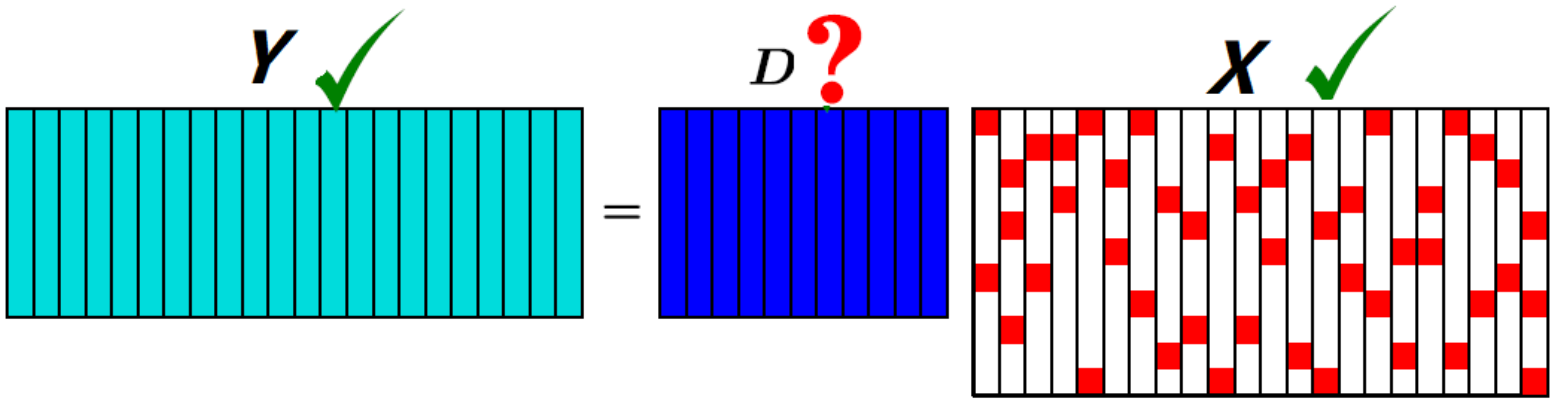
Figure: Classification accuracies of different sparse coding algorithms with different training samples with different numbers of training samples

Summary

- Sparse coding is an NP-hard problem.
- **Greedy algorithms** provide an approximate solution.
  - MP: Iteratively choose the best atom from the dictionary.
  - OMP: MP $+$ full backward orthogonality of the residual.
- **Relaxation algorithms** use l_1 norm to guarantee sparsity.
  - ISTA: construct proximal mapping via Majorization-Minimization idea.
  - FISTA: use momentum idea to accelerate ISTA.
  - Coordinate Descent: successively minimizes along coordinate directions to find the minimum.

Outline

Algorithms of Dictionary Learning



Dictionary Learning: Fix X, and update D

$$\min_{D} \|Y - DX\|_F^2$$

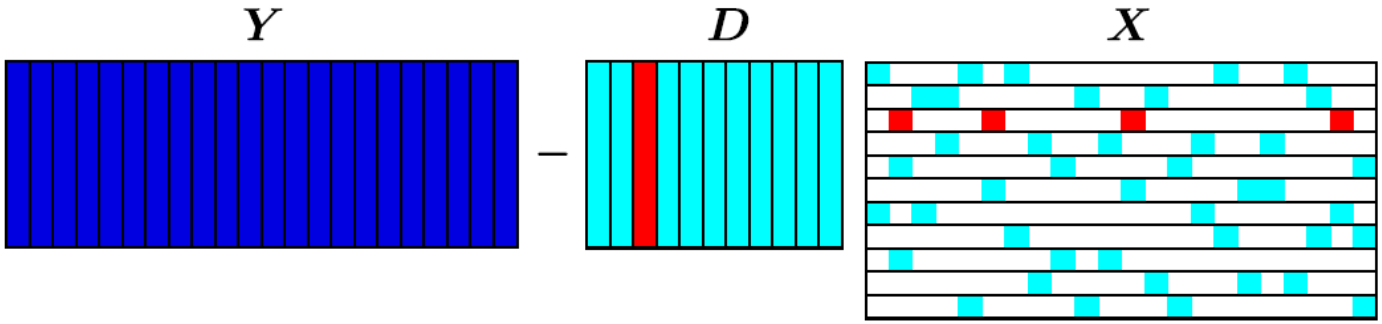
- Method of Optimal Directions (MOD) [Engan et al., 1999]
- K-SVD [Aharon et al., 2006]

Methods of Optimal Directions

- MOD alternates between getting the sparse coding and updating the dictionary by computing the analytical solution.
- Sparse coding: MP, OMP, ISTA, ...
- Dictionary learning: $D = YX^{+}$, where X^{+} is a Moore-Penrose pseudoinverse.
- Due to the high complexity of the matrix-inversion operation, computing the pseudoinverse is intractable in many cases.

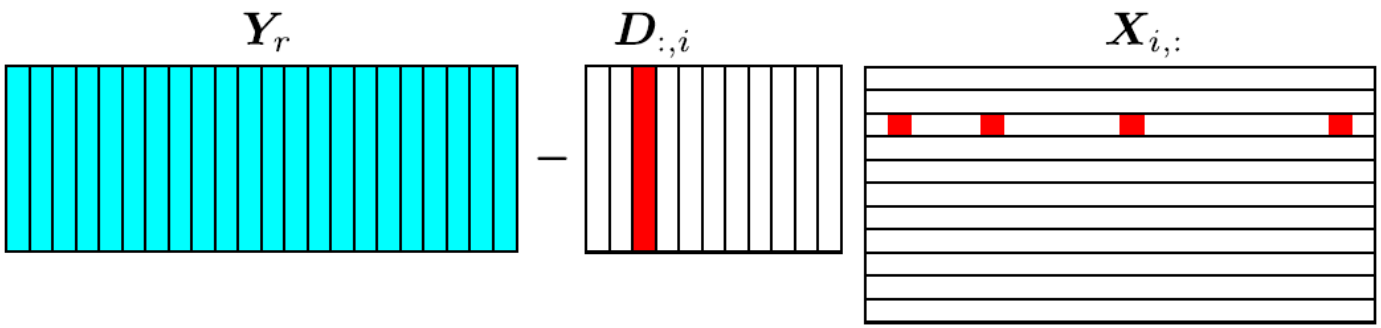
K-SVD

$$\|Y - DX\|_F^2$$
$$= \|Y - D_{:j\neq i}X_{j\neq i,i} - D_{:,i}X_{i,:}\|_F^2$$



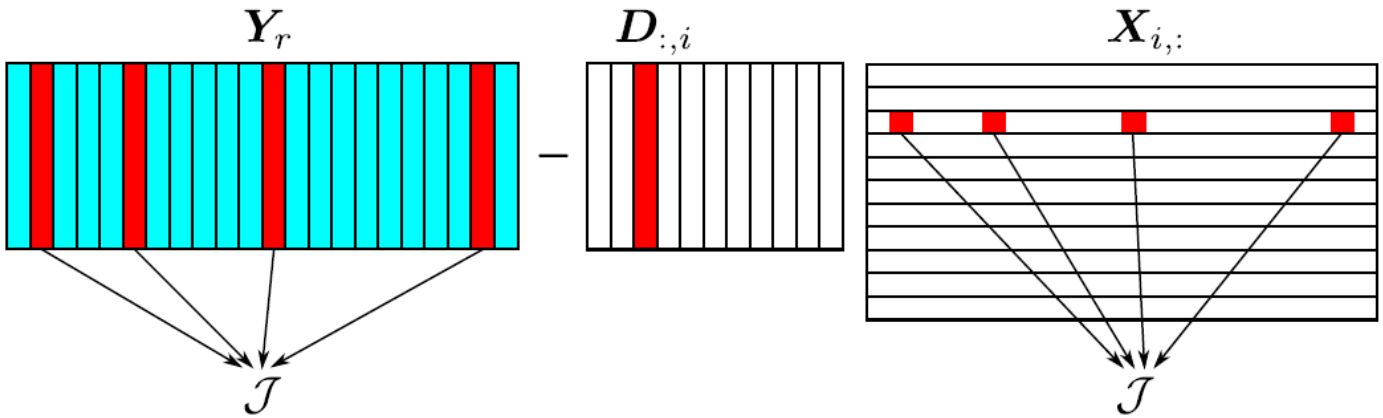
K-SVD

$$
\begin{aligned}
&\|Y - DX\|_F^2 \\
&= \|Y - D_{:j\neq i}X_{j\neq i,i} - D_{:,i}X_{i,:}\|_F^2 \\
&= \|Y_r - D_{:,i}X_{i;}\|_F^2
\end{aligned}
$$



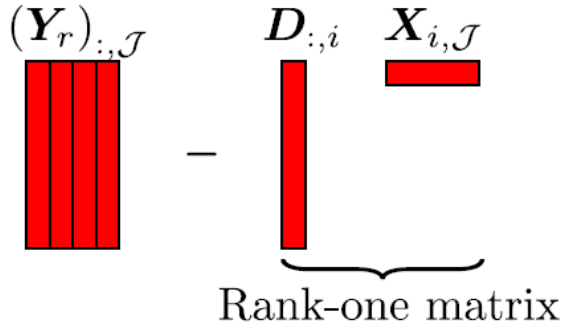
K-SVD

$$
\begin{aligned}
&\|Y - DX\|_F^2 \\
&= \|Y - D_{:j\neq i}X_{j\neq i,i} - D_{:,i}X_{i,:}\|_F^2 \\
&= \|Y_r - D_{:,i}X_{i;}\|_F^2 \\
&= \left\|(Y_r)_{:,\mathcal{J}} - D_{:,i}X_{i,\mathcal{J}}\right\|_F^2 + c
\end{aligned}
$$



K-SVD

$$
\begin{aligned}
&\|Y - DX\|_2^2 \\
&= \|Y - D_{:j\neq i}X_{j\neq i,i} - D_{:,i}X_{i,:}\|_2^2 \\
&= \|Y_r - D_{:,i}X_{i;}\|_2^2 \\
&= \left\|(Y_r)_{:,\mathcal{J}} - D_{:,i}X_{i,\mathcal{J}}\right\|_2^2 + c
\end{aligned}
$$



SVD: optimal rank-one matrix approximation:

$$
\begin{aligned}
A &= \sum \lambda_i \boldsymbol{u}_i \boldsymbol{v}_i^T \\
&\approx \lambda_1 \boldsymbol{u}_1 \boldsymbol{v}_1^T
\end{aligned}
\qquad \lambda_1 \geqslant \lambda_2 \geqslant \cdots \geqslant \lambda_n
$$

K-SVD

Task: Find the best dictionary to represent the data samples $\{\mathbf{y}_i\}_{i=1}^N$ as sparse compositions, by solving

$$\min_{\mathbf{D},\mathbf{X}} \{\|\mathbf{Y} - \mathbf{D}\mathbf{X}\|_F^2\} \quad \text{subject to} \quad \forall i,\ \|\mathbf{x}_i\|_0 \le T_0.$$

Initialization : Set the dictionary matrix $\mathbf{D}^{(0)} \in \mathbb{R}^{n\times K}$ with ℓ^2 normalized columns. Set $J = 1$.
Repeat until convergence (stopping rule):

- *Sparse Coding Stage*: Use any pursuit algorithm to compute the representation vectors $\mathbf{x}_i$ for each example $\mathbf{y}_i$, by approximating the solution of

$$i = 1, 2, \ldots,\ N, \quad \min_{\mathbf{x}_i} \{\|\mathbf{y}_i - \mathbf{D}\mathbf{x}_i\|_2^2\} \quad \text{subject to} \quad \|\mathbf{x}_i\|_0 \le T_0.$$

- *Codebook Update Stage*: For each column $k = 1, 2, \ldots, K$ in $\mathbf{D}^{(J-1)}$, update it by
  - Define the group of examples that use this atom, $\omega_k = \{i \mid 1 \le i \le N,\ \mathbf{x}_T^k(i) \ne 0\}$.
  - Compute the overall representation error matrix, $\mathbf{E}_k$, by

$$\mathbf{E}_k = \mathbf{Y} - \sum_{j\ne k} \mathbf{d}_j \mathbf{x}_T^j.$$

  - Restrict $\mathbf{E}_k$ by choosing only the columns corresponding to ω_k, and obtain $\mathbf{E}_k^R$.
  - Apply SVD decomposition $\mathbf{E}_k^R = \mathbf{U}\mathbf{\Delta}\mathbf{V}^T$. Choose the updated dictionary column $\tilde{\mathbf{d}}_k$ to be the first column of $\mathbf{U}$. Update the coefficient vector $\mathbf{x}_R^k$ to be the first column of $\mathbf{V}$ multiplied by $\mathbf{\Delta}(1,1)$.
- Set $J = J + 1$.

Atoms Learned by K-SVD

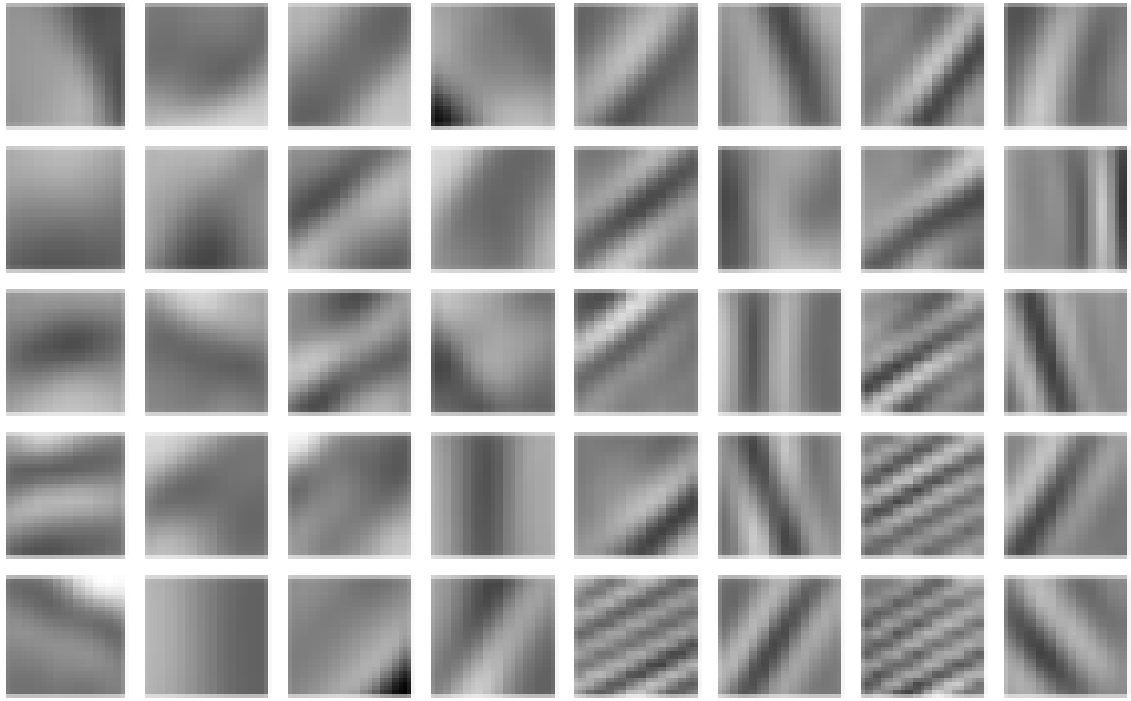


Figure: Atoms from a K-SVD dictionary trained on 12×12 image patches from Lena

MOD vs K-SVD

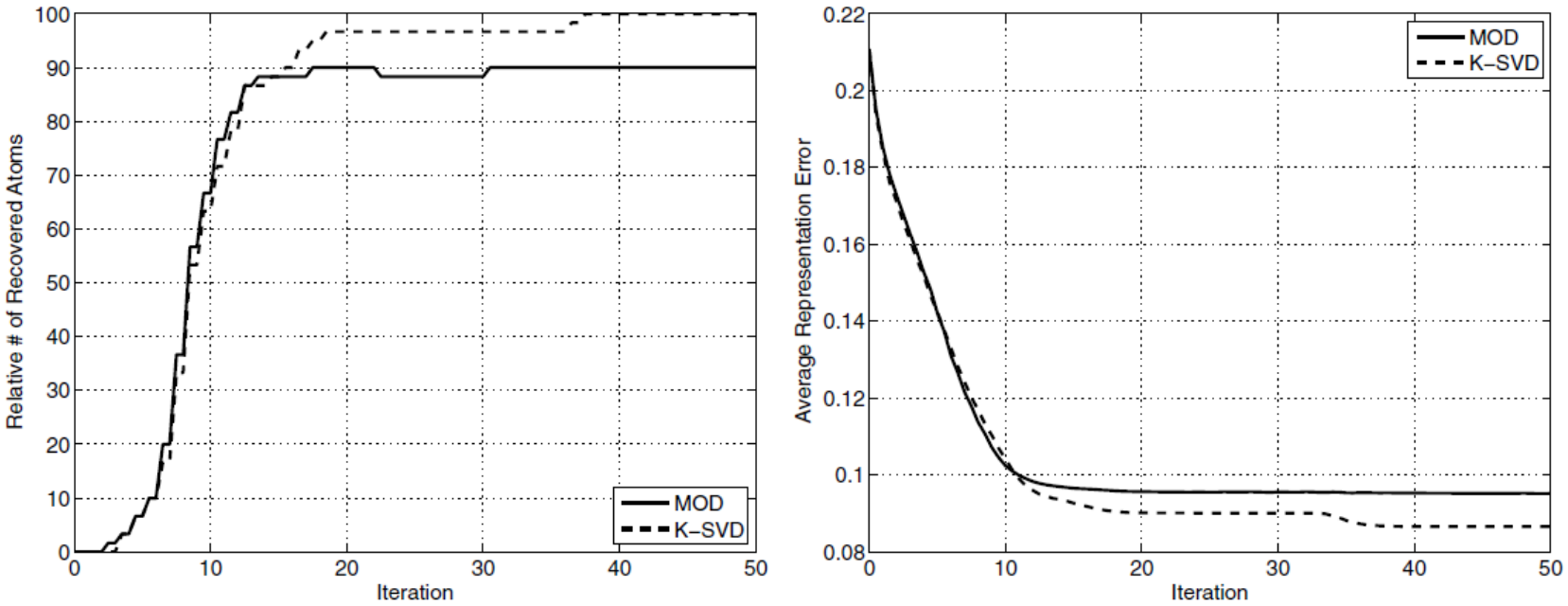


Figure: The K-SVD behavior in a synthetic experiment. The left graph presents the relative number (in %) of correctly recovered atoms, and the right one shows the average representation error.

Theoretical Foundations of Dictionary Learning

- We have seen that pursuit algorithms enjoy theoretical guarantees for their success. Could one offer a similar concept for Dictionary Learning?

Theoretical Foundations of Dictionary Learning

- We have seen that pursuit algorithms enjoy theoretical guarantees for their success. Could one offer a similar concept for Dictionary Learning?

- Imagine a "Theorem" like this:
  Theorem: Given a set of N ϵ-noisy training examples originated from Sparse Coding Model, having each k_0-sparse representation w.r.t. to a given dictionary D, then under certain conditions, the algorithm is guaranteed to stably recover D

Theoretical Foundations of Dictionary Learning

- We have seen that pursuit algorithms enjoy theoretical guarantees for their success. Could one offer a similar concept for Dictionary Learning?

- Imagine a "Theorem" like this:
  Theorem: Given a set of N ϵ-noisy training examples originated from Sparse Coding Model, having each k_0-sparse representation w.r.t. to a given dictionary D, then under certain conditions, the algorithm is guaranteed to stably recover D

- Various attempts have been made to form such theorems, under different assumptions and/or with different claims.

Theoretical Foundations of Dictionary Learning

Several representative studies:

- New Algorithms for Learning Incoherent and Overcomplete Dictionaries [Arora et al., 2014].
- Local Identifiability of Dictionary Learning [Schnass, 2015].
- Complete Dictionary Recovery Over the Sphere I: Overview and the Geometric Picture [Sun et al., 2016].
- Dictionary Learning With Few Samples and Matrix Concentration [Luh and Vu, 2016].
- Local Identifiability of l_1-minimization Dictionary Learning: a Sufficient and Almost Necessary Condition [Wu and Yu, 2017].

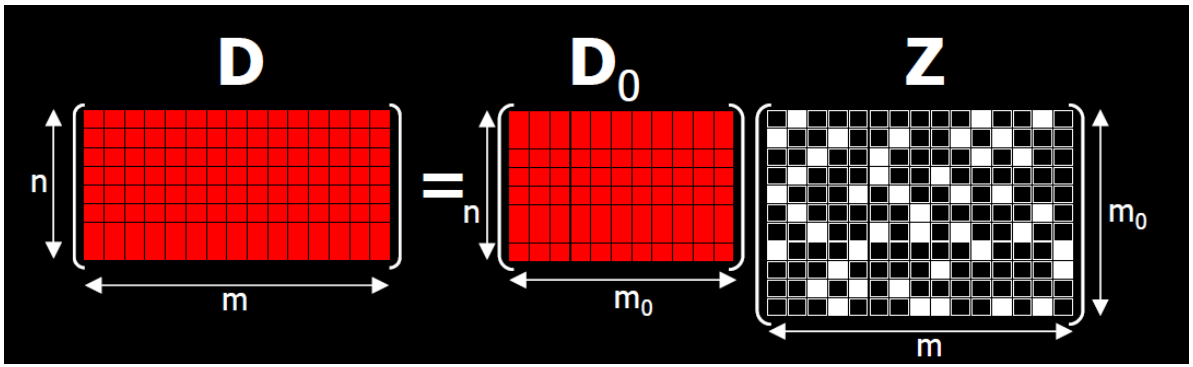
Dictionary Learning: Main Difficulties

- Dictionary Learning is a great idea, leading to a major boost in the development of sparse coding.
- However, dictionary Learning has some major problems:
  - Speed and memory problems
  - Restriction to low-dimensions
- Several studies have been proposed to solve this problems:
  - Double-Sparsity Dictionary [Rubinstein et al., 2009]
  - Union of Unitary Dictionary [Lesage et al., 2005]

Double-Sparsity Dictionary

The Basic Idea:

- Assume that the dictionary to be found D can be written as $D = D_0 Z$, where:
  - D_0 is a fixed base-dictionary with a fast deployment
  - Z is a very sparse matrix, having $k_1 \ll n$ non-zeros in each of its columns
- This structure assumes that each atom in D can be described as a sparse combination of atoms from D_0



Double-Sparsity Dictionary

$$\min_{Z,\boldsymbol{x}} \|\boldsymbol{y} - D_0 Z\boldsymbol{x}\|_2^2$$
$$\text{s.t.} \begin{cases} \|Z_i\|_0 \leqslant t \\ \|\boldsymbol{x}\|_0 \leqslant p \end{cases}$$

Benefits:

- Multiplying by D (and its transpose) is fast, since the multiplication by D_0 is fast, and multiplication by a sparse matrix is cheap.
- The free degree is small ($2mk_1$ instead of mn), less examples are needed for training and better convergence could be obtained.
- Higher-dimension signals can be treated.

Union of Unitary Dictionary

What if D is required to be square and unitary?

- Pursuit becomes easy

$$\min_{\boldsymbol{x}} \|\boldsymbol{y} - D\boldsymbol{x}\|_2^2 \quad \text{s.t. } \|\boldsymbol{x}\|_0 \leqslant k_0 \Rightarrow \boldsymbol{x} = \mathcal{H}_{k_0}(D^T\boldsymbol{y})$$

- The free degree of D is only $n(n-1)/2$, this implies less examples to train on, better generalization, and better convergence

What about Overcomplete Dictionary?

- Union of Unitary Dictionary!

$$\min_{D_1, D_2, \boldsymbol{x}_1, \boldsymbol{x}_2} \|\boldsymbol{y} - D_1\boldsymbol{x}_1 - D_2\boldsymbol{x}_2\|_2^2$$

$$\text{s.t.} D_1^T D_1 = I,\ D_2^T D_2 = I,\ \|\boldsymbol{x}_1\|_0 \leqslant k,\ \|\boldsymbol{x}_2\|_0 \leqslant k$$

Summary

- Dictionary Learning: fix X, and update D
  - MOD: updating the dictionary by computing the analytical solution of the problem given by $D = YX^{+}$
  - K-SVD: atom-by-atom in a simple and efficient process
- There are several theoretical foundations guaranteeing the recovery of dictionary.
- Several studies have been put forward to speed up dictionary learning and adapt to high dimension:
  - Double-Sparsity Dictionary
  - Union of Unitary Dictionary

Outline

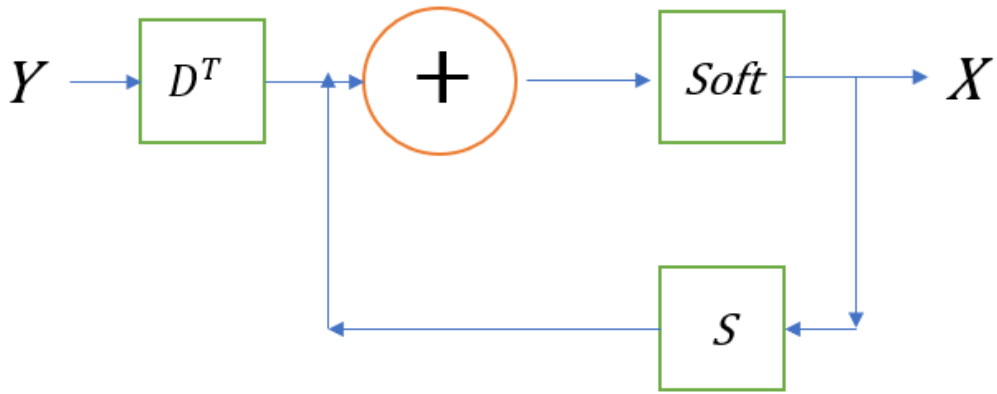
Learned ISTA

Review ISTA:

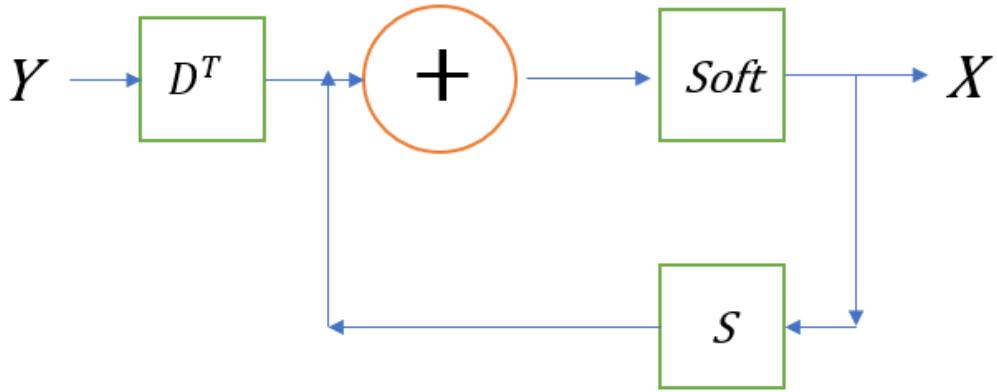
Algorithm 5 ISTA

1: Input: Signal $\boldsymbol{y}$, normalized dictionary D.
2: Initialization: $\boldsymbol{x} = \mathbf{0}, L > \sigma_D$;
3: **while** above the threshold **do**
4: $\quad \boldsymbol{x}^{(k+1)} = S_{(\lambda/L)}\left(\boldsymbol{x}^{(k)} - \frac{1}{L}D^T\left(D\boldsymbol{x}^{(k)} - \boldsymbol{y}\right)\right)$
5: **end while**

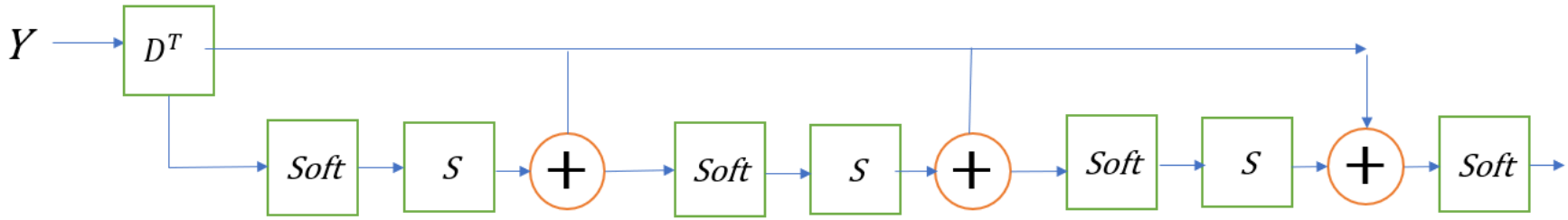
The workflow of ISTA can be represented as:



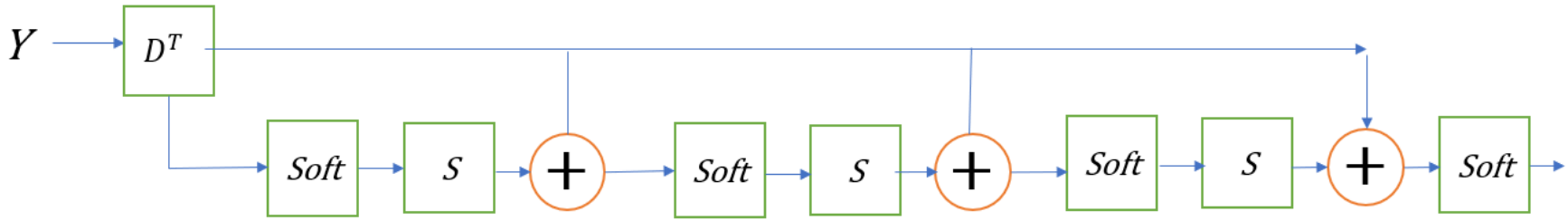
Learned ISTA [Gregor and LeCun, 2010]



"Learned ISTA", uses a time unfolded version of the ISTA block diagram, truncated to a fixed number of iterations (3 here):



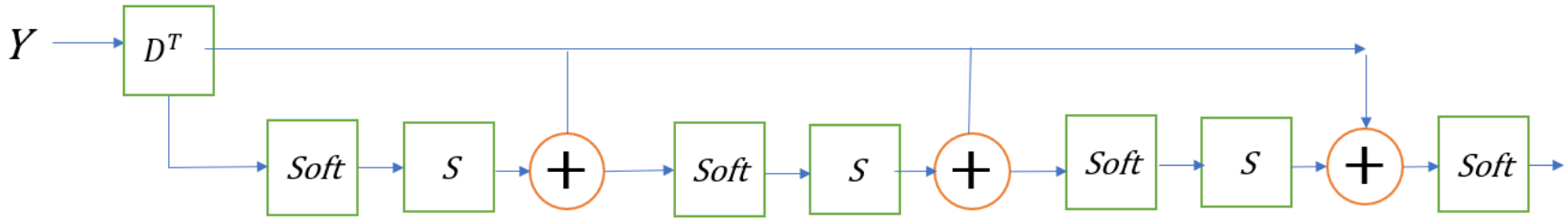
Learned ISTA (LISTA)



- Inspired by ISTA, we unfold the ISTA and design a recurrent neural network to approximate sparse coding.

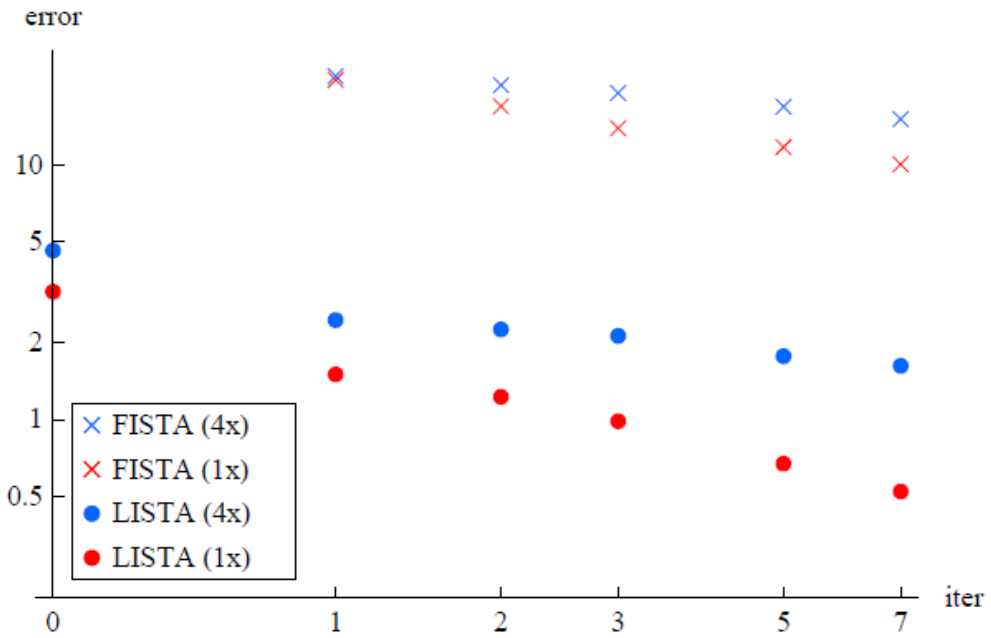
- Two question raised:
  - How to train the network?
  - Does LISTA really work?

How to Train the Network?



- Training set: $Y = [\boldsymbol{y}_1, \cdots, \boldsymbol{y}_p]$
- Label: $X^* = [\boldsymbol{x}_1^*, \cdots, \boldsymbol{x}_p^*]$, is the optimal code for sample Y, as obtained with the Coordinate Descent method.
- Loss function: $\mathcal{L}(W, Y) = \frac{1}{2} \|X^* - f_e(W, Y)\|_F^2$
- Optimizer: SGD

Does LISTA Really Work?



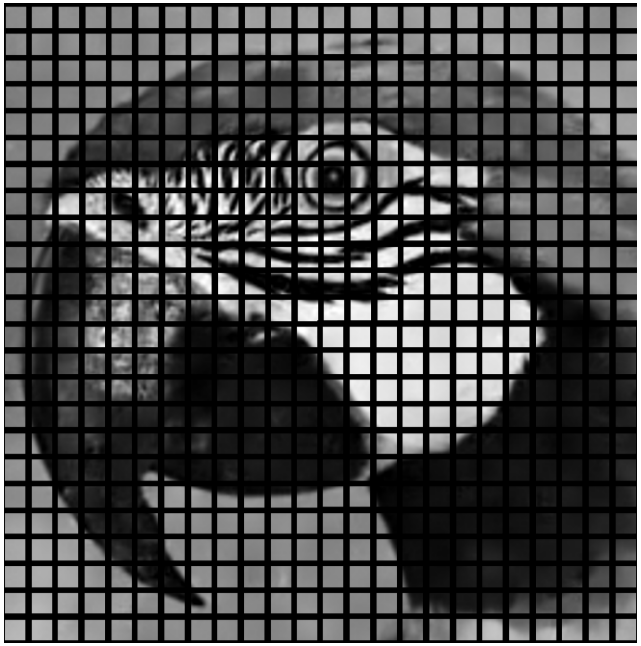
- Code prediction error as a function of number of iterations for FISTA (crosses) and for LISTA (dots), for $m = 100$ (red) and $m = 400$ (blue), where m is the number of atoms in dictionary.
- It takes 18 iterations of FISTA to reach the error of LISTA with just one iteration for $m = 100$, and 35 iteration for $m = 400$.

Learned ISTA

- **Main idea:** Unfold ISTA and design a new structure of neural network to train sparse coding.

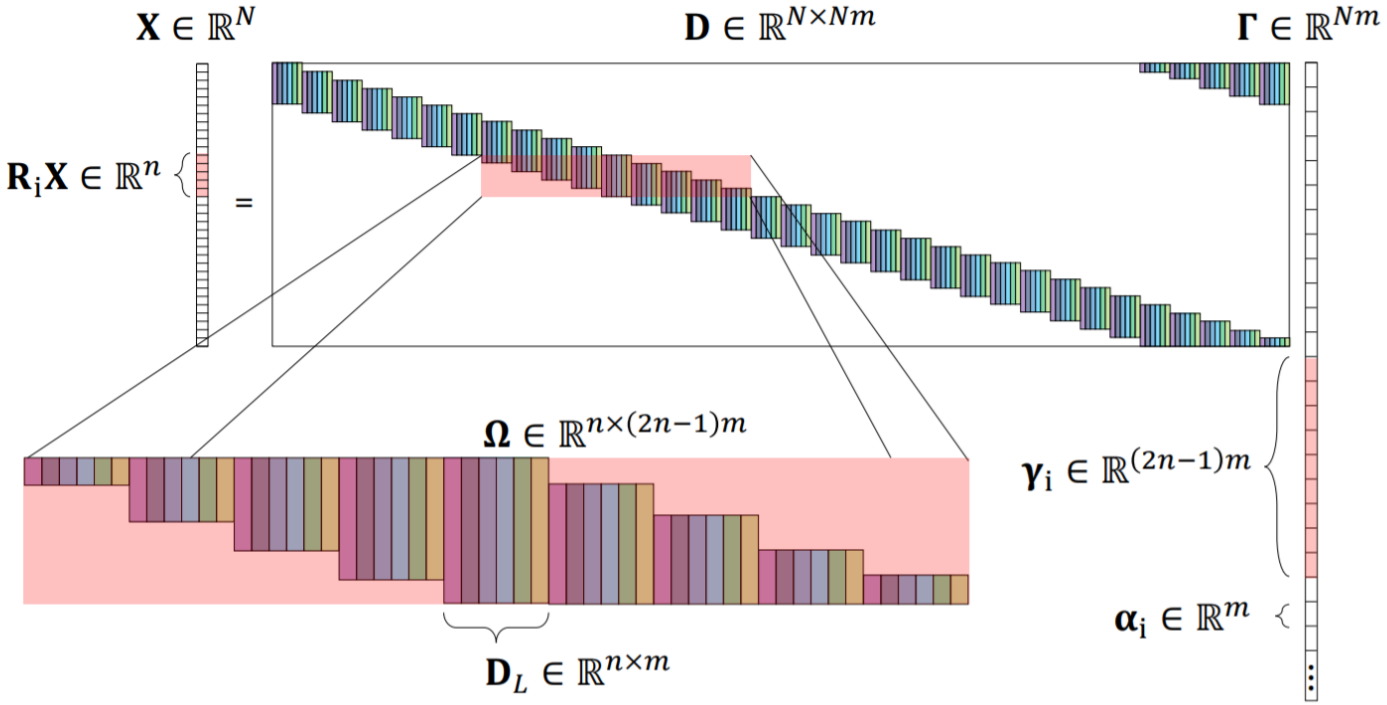
- **Main contributions:**
  - Propose fast encoders that can be trained to compute approximate sparse codes.
  - Firstly use neural network to improve traditional algorithms.

Convolutional Sparse Coding (CSC) [Zeiler et al., 2010]



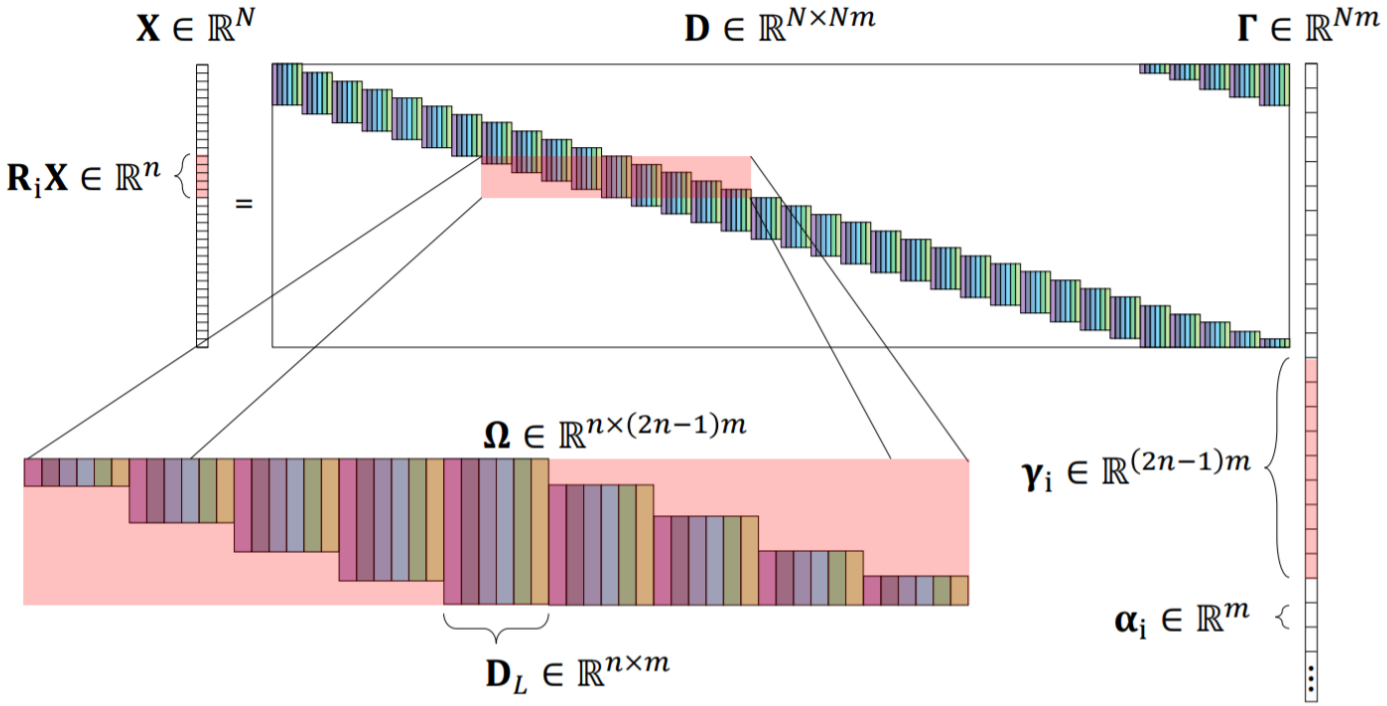
- For many natural signals, however, sparse coding is applied to sub-elements (i.e. patches) of the signal, where such an assumption is invalid.
- Convolutional sparse coding explicitly models local interactions through the convolution operator.

The CSC Model and Its Constituent Elements



In the following topic, we will use $\boldsymbol{X}$ and $\boldsymbol{\Gamma}$ to represent a signal and its representation respectively. Also we defined $R_i \in \mathbb{R}^{n \times N}$ to be the operator that extracts the i-th n-dimensional patch from $\boldsymbol{X}$.

CSC Dictionary Learning via Local Processing



$$\min_{D_L, \{\boldsymbol{\alpha}_i\}_{i=1}^{N=1}, \{\boldsymbol{s}_i\}_{i=1}^{N=1}} \frac{1}{2} \left\| \boldsymbol{X} - \sum_{i=1}^{N} R_i^T \boldsymbol{s}_i \right\|_2^2 + \lambda \sum_{i=1}^{N} \|\boldsymbol{\alpha}_i\|_1$$

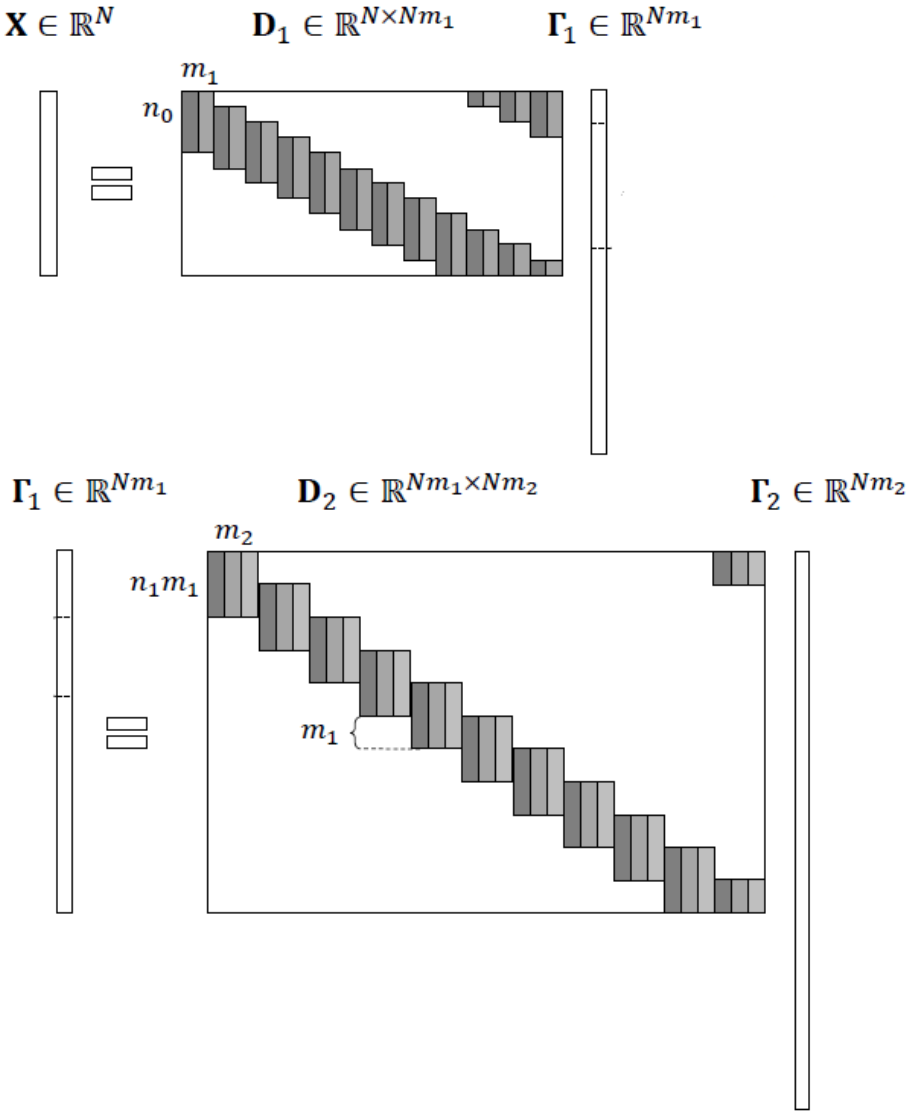
$$\text{s.t. } \boldsymbol{s}_i = D_L \boldsymbol{\alpha}_i$$

CSC Dictionary Learning via Local Processing

The main advantages:

- It operates locally on patches, while solving faithfully the global CSC problem;
- It can leverage standard techniques from the sparse representations field, such as OMP, LARS, K-SVD, MOD, online dictionary learning and trainlets;
- When compared to state-of-the-art methods, it can be applied to standard-sized images, converges faster and provides a better model;

Multi-Layered CSC



Multi-Layered CSC

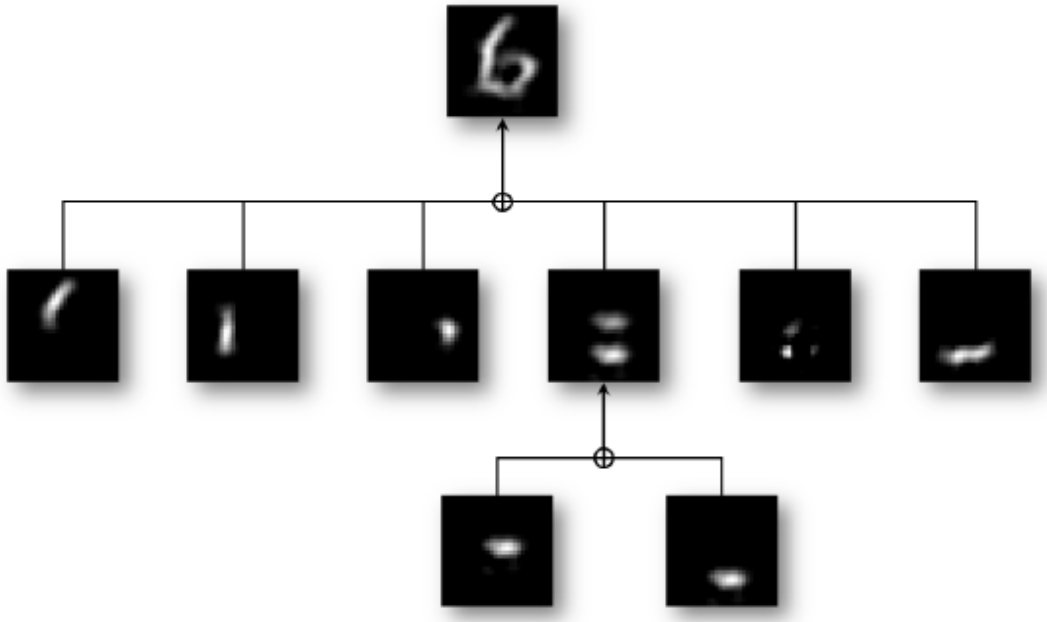


Figure: From atoms to molecules: Illustration of the ML-CSC model for a number 6.

Multi-Layered CSC

Motivation and Goals:

- Convolutional neural networks (CNN) lead to remarkable results in many fields.
  - Clear and profound theoretical understanding is still lacking.

Multi-Layered CSC

Motivation and Goals:

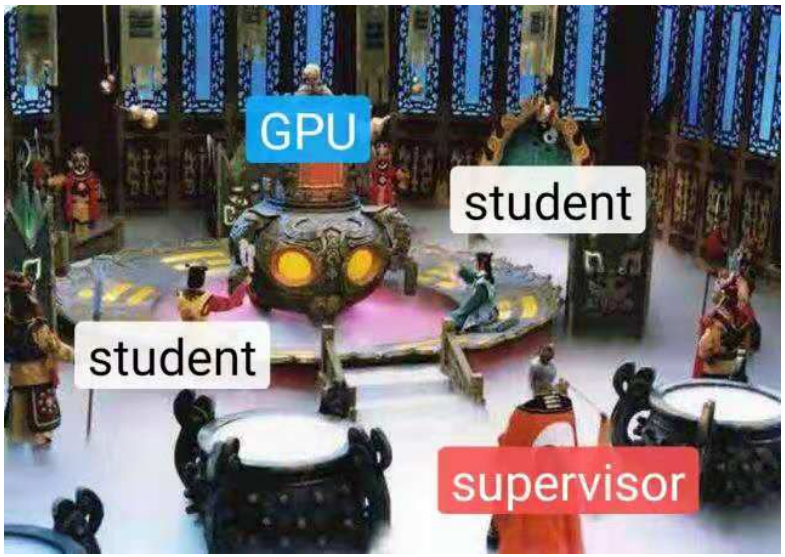
- Convolutional neural networks (CNN) lead to remarkable results in many fields.
  - Clear and profound theoretical understanding is still lacking.

- Sparse representation is a powerful model
  - Enjoys from a vast theoretical study, supporting its success.
  - Recently, convolutional sparse coding (CSC) has also been analyzed thoroughly.

Multi-Layered CSC

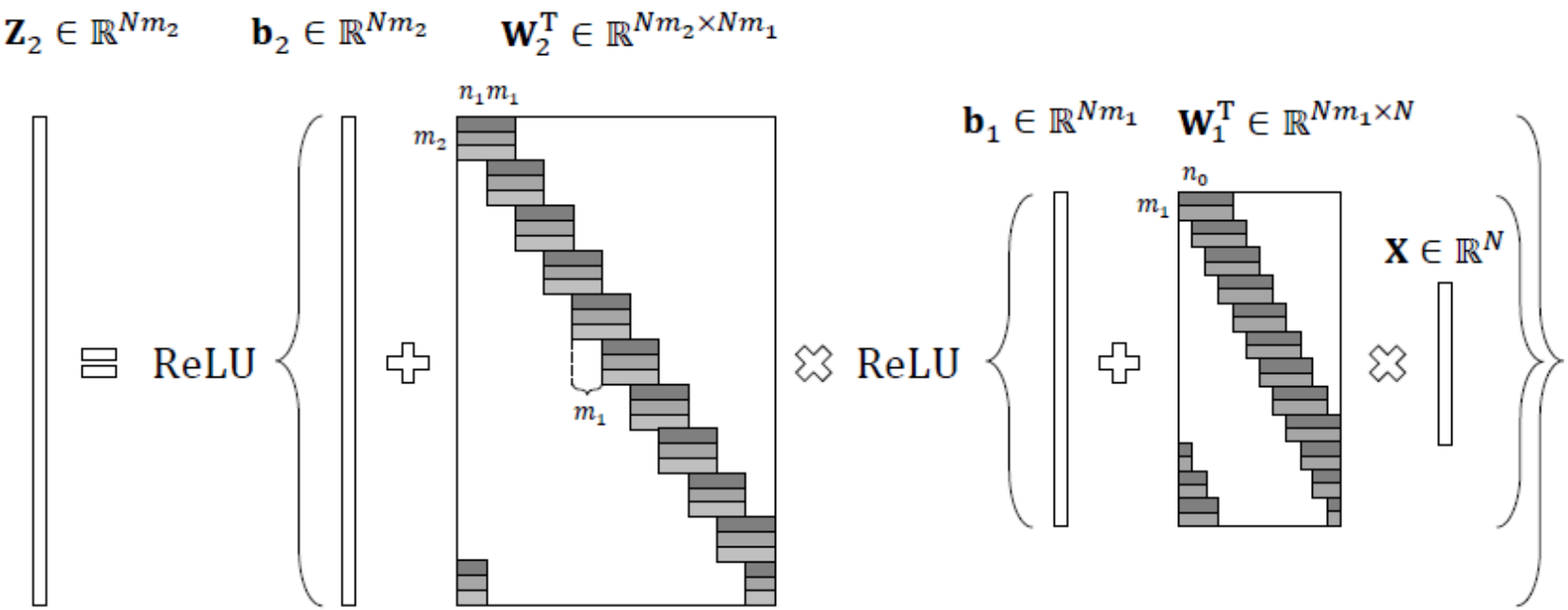
Motivation and Goals:

- There seems to be a relation between CSC and CNN:
  - Both have a convolutional structure
  - Both use a data driven approach for training the model
  - The most popular non-linearity employed in CNN, called ReLU, is known to be connected to sparsity (and shrinkage)



Can we build some theoretical basis of CNN via multi-layered CSC?

CNN: The Forward Pass



$$f(\boldsymbol{X}) = \mathrm{ReLU}\left(\boldsymbol{b}_2 + W_2^T \,\mathrm{ReLU}\left(\boldsymbol{b}_1 + W_1^T \boldsymbol{X}\right)\right)$$

Multi-Layered CSC [Papyan et al., 2017]

Consider the ML-CSC model defined by the set of dictionaries $\{D_i\}_{i=1}^K$, and given a signal $\boldsymbol{X}$

$$\begin{aligned} \boldsymbol{X} &= D_1\Gamma_1 \quad && \|\Gamma_1\|_{0,\infty}^s \leqslant \lambda_1 \\ \Gamma_1 &= D_2\Gamma_2 \quad && \|\Gamma_2\|_{0,\infty}^s \leqslant \lambda_2 \\ & && \cdots \\ \Gamma_{K-1} &= D_K\Gamma_K \quad && \|\Gamma_K\|_{0,\infty}^s \leqslant \lambda_K \end{aligned}$$

where $\|\Gamma\|_{0,\infty}^s \triangleq \max_i \|\gamma_i\|_0$

Multi-Layered CSC

Layered Thresholding: a approximated algorithm to solve multi-layered convolutional sparse coding.

Algorithm 1 The layered thresholding algorithm.

Input:

$\mathbf{X}$ – a signal.
$\{\mathbf{D}_i\}_{i=1}^K$ – convolutional dictionaries.
$\mathcal{P} \in \{\mathcal{H}, \mathcal{S}, \mathcal{S}^+\}$ – a thresholding operator.
$\{\beta_i\}_{i=1}^K$ – thresholds.

Output:

A set of representations $\{\hat{\mathbf{\Gamma}}_i\}_{i=1}^K$.

Process:

1: $\hat{\mathbf{\Gamma}}_0 \leftarrow \mathbf{X}$
2: **for** $i = 1 : K$ **do**
3: $\quad \hat{\mathbf{\Gamma}}_i \leftarrow \mathcal{P}_{\beta_i}(\mathbf{D}_i^T \hat{\mathbf{\Gamma}}_{i-1})$
4: **end for**

Multi-Layered CSC and CNN

Consider 2-layered CSC problem and 2-layer CNN:

Layered Thresholding:

$$\boldsymbol{\Gamma}_2 = \mathcal{P}_{\beta_2}\left(D_2^T \mathcal{P}_{\beta_1}\left(D_1^T \boldsymbol{X}\right)\right)$$

The Forward Pass of CNN:

$$f(\boldsymbol{X}) = \text{ReLU}\left(\boldsymbol{b}_2 + W_2^T \text{ReLU}\left(\boldsymbol{b}_1 + W_1^T \boldsymbol{X}\right)\right)$$

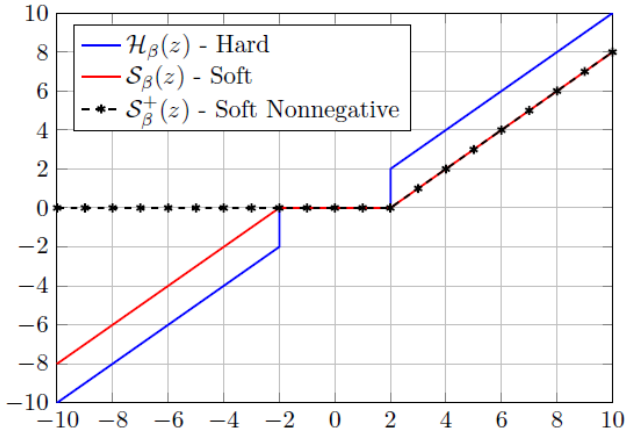


Figure 3: The thresholding operators for a constant $\beta = 2$.

The layered (soft nonnegative) thresholding and the forward pass algorithm are the same!!!

ML-CSC: Pursuit and Dictionary Learning

Multi-Layer Convolutional Sparse Modeling [Sulam et al., 2018]: Given a set of convolutional dictionaries $\{D_i\}_{i=1}^{L}$ of appropriate dimensions, a signal $\boldsymbol{X}\left(\boldsymbol{\Gamma}_i\right) \in \mathbb{R}^N$ admits a representation in terms of the ML-CSC model, i.e. $\boldsymbol{X}\left(\boldsymbol{\Gamma}_i\right) \in \mathcal{M}_{\boldsymbol{\lambda}}$, if:

$$
\begin{aligned}
\boldsymbol{X} &= D_1\boldsymbol{\Gamma}_1, \left\|\boldsymbol{\Gamma}_1\right\|_{0,\infty}^{s} \leqslant \lambda_1 \\
\boldsymbol{\Gamma}_1 &= D_2\boldsymbol{\Gamma}_2, \left\|\boldsymbol{\Gamma}_2\right\|_{0,\infty}^{s} \leqslant \lambda_2 \\
&\vdots \\
\boldsymbol{\Gamma}_{L-1} &= D_L\boldsymbol{\Gamma}_L, \left\|\boldsymbol{\Gamma}_L\right\|_{0,\infty}^{s} \leqslant \lambda_L
\end{aligned}
$$

Two main steps to solve the model:

- Basis Pursuit
- Dictionary Learning

Basis Pursuit for ML-CSC

- Layered Thresholding
  - Use thresholding function in a layer-wise manner
  - The same as the forward pass of a CNN
  - Stability guarantees can be proved under certain conditions
- ADMM [Sulam et al., 2019]
- Multi-Layered ISTA [Sulam et al., 2019]

Dictionary Learning for ML-CSC

Sparse Dictionaries Assumption [Sulam et al., 2018]:

- If both Γ_L and $\Gamma_{L-1} = D_L\Gamma_L$ are sparse, then atoms in D must indeed contain only a few non-zeros.

- Minimizing the sparsity of any i^{th} representation can be done implicitly by minimizing the sparsity of the last layer and the number of non-zeros in the dictionaries.

$$\|\Gamma_i\|_{0,\infty}^s \leqslant c \prod_{j=i+1}^{L} \|D_j\|_0 \, \|\Gamma_L\|_{0,\infty}^s$$

Dictionary Learning for ML-CSC

With the sparse dictionary assumption, we recast the primary problem into the following problem:

$$\min_{\left\{\Gamma_L^k\right\},\{D_i\}} \sum_{k=1}^{K} \left\| \boldsymbol{X}^k - D_1 D_2 \dots D_L \boldsymbol{\gamma}_L^k \right\|_2^2 + \sum_{i=2}^{L} \zeta_i \left\| D_i \right\|_0$$

$$\text{s.t.} \begin{cases} \left\| \Gamma_L^k \right\|_{0,\infty}^s \leqslant \lambda_L \\ \left\| \boldsymbol{d}_i^j \right\|_2 = 1, \forall i, j \end{cases}$$

Consider the unconstrained form:

$$\min_{\left\{\Gamma_L^k\right\},\{D_i\}} \sum_{k=1}^{K} \left\| \boldsymbol{X}^k - D_1 D_2 \dots D_L \Gamma_L^k \right\|_2^2 + \iota \sum_{i=1}^{L} \left\| D_i \right\|_F^2 + \sum_{i=2}^{L} \zeta_i \left\| D_i \right\|_0 + \lambda \left\| \Gamma_L^k \right\|_1$$

Dictionary Learning for ML-CSC

$$\min_{\{\Gamma_L^k\},\{D_i\}} \sum_{k=1}^{K} \left\| \boldsymbol{X}^k - D_1 D_2 \dots D_L \Gamma_L^k \right\|_2^2 + \iota \sum_{i=1}^{L} \left\| D_i \right\|_F^2 + \sum_{i=2}^{L} \zeta_i \left\| D_i \right\|_0 + \lambda \left\| \Gamma_L^k \right\|_1$$

Algorithm 3: Multi-Layer Convolutional Dictionary Learning

Data: Training samples $\{\boldsymbol{X}_k\}_{k=1}^K$, initial convolutional dictionaries $\{\mathbf{D}_i\}_{i=1}^L$

for $k = 1, \dots, K$ **do**

Draw $\boldsymbol{X}_k$ at random;

Sparse Coding:

$\gamma_L \leftarrow \arg\min_{\gamma} \|\boldsymbol{X}_k - \mathbf{D}^{(L)}\gamma\|_2 + \lambda\|\gamma\|_1$;

Update Dictonaries:

for $i = L, \dots, 2$ **do**

for $t = 1, \dots, T$ **do**

$\mathbf{D}_i^{t+1} \leftarrow \mathcal{H}_{\zeta_i}\left[\mathbf{D}_i^t - \eta \nabla f(\mathbf{D}_i^t)\right]$;

for $t = 1, \dots, T$ **do**

$\mathbf{D}_1^{t+1} \leftarrow \mathbf{D}_1^t - \eta \nabla f(\mathbf{D}_1^t)$;

ML-CSC: Pursuit and Dictionary Learning

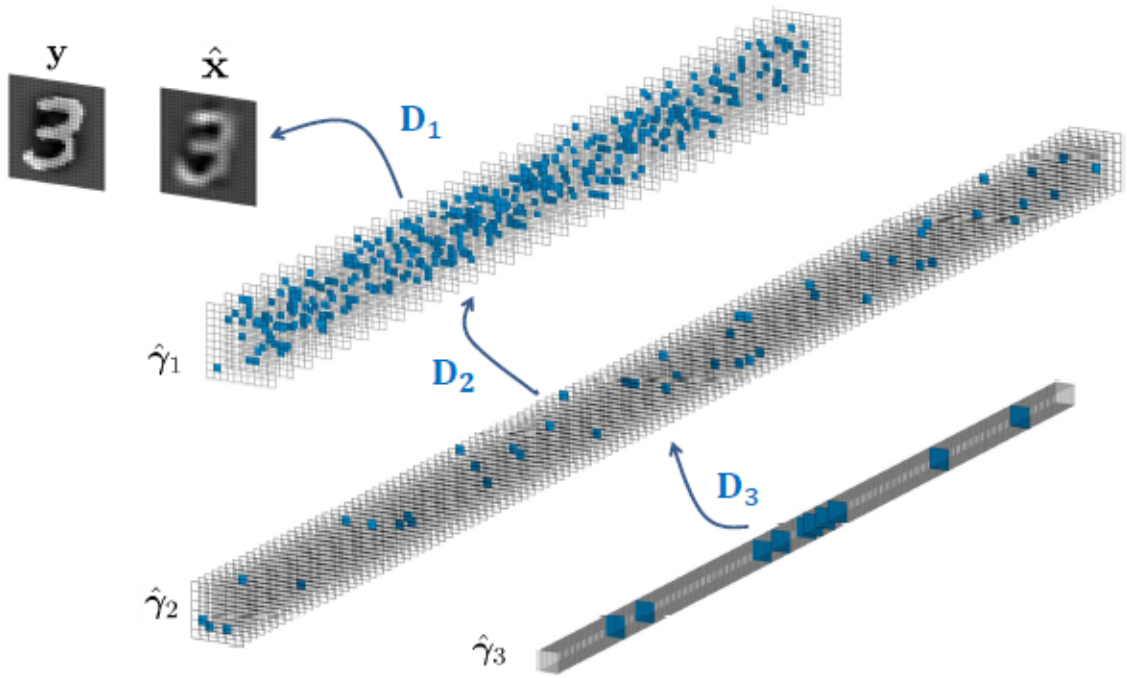


Figure: Decompositions of an image from MNIST in terms of its nested sparse features γ_i and multi-layer convolutional dictionaries D_i.

ML-CSC: Pursuit and Dictionary Learning

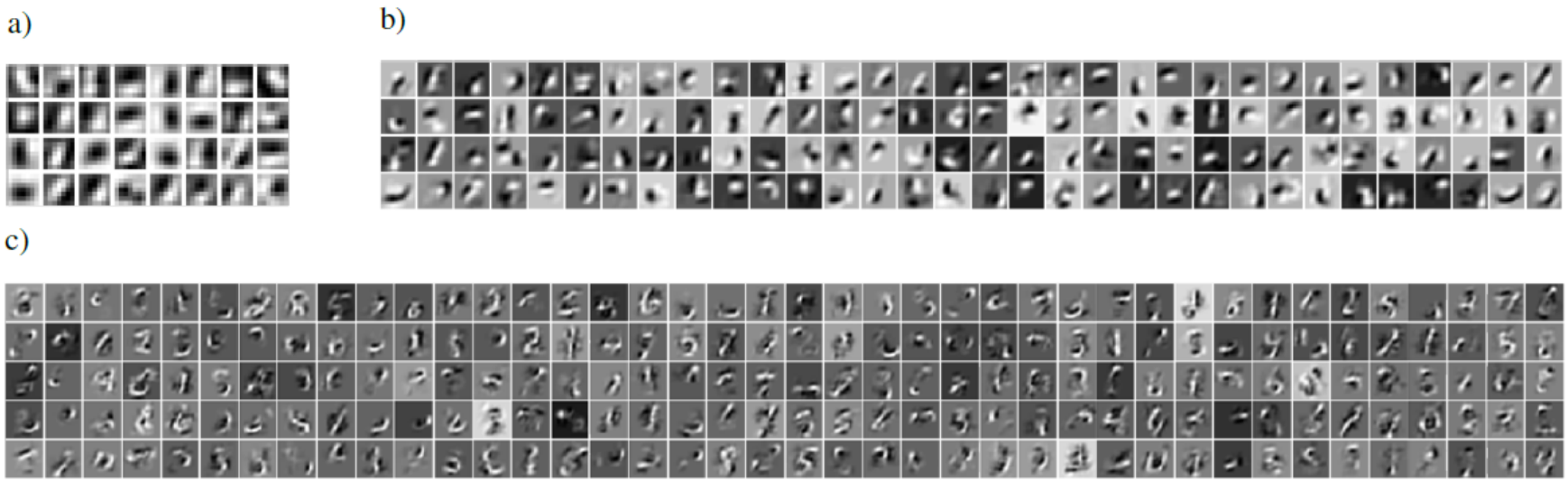


Figure: ML-CSC model trained on the MNIST dataset, a) The local filters of the dictionary D_1. b) The local filters of the effective dictionary $D^{(2)} = D_1 D_2$. c) Some of the 1024 local atoms of the effective dictionary $D^{(3)}$.

Summary

- Learned ISTA
  - Unfold ISTA and design a new structure of neural network to train sparse coding
  - Firstly use neural network to improve traditional algorithms
- Convolutional Sparse Coding
  - Why Convolutional? **Local interactions!**
  - Dictionary can be learned via local processing
- Multi-Layered CSC
  - Why Deep? **Learn more complex filters!**
  - Related closely with CNN
  - Sparse dictionaries assumption

References I


Aharon, M., Elad, M., and Bruckstein, A. (2006).
K-svd: An algorithm for designing overcomplete dictionaries for sparse representation.
IEEE Transactions on signal processing, 54(11):4311–4322.


Arora, S., Ge, R., and Moitra, A. (2014).
New algorithms for learning incoherent and overcomplete dictionaries.
In *Conference on Learning Theory*, pages 779–806.


Barlow, H. B. (1981).
Critical limiting factors in the design of the eye and visual cortex.
Proc. R. Soc. Lond., B, 212:1–34.


Beck, A. and Teboulle, M. (2009).
A fast iterative shrinkage-thresholding algorithm for linear inverse problems.
SIAM journal on imaging sciences, 2(1):183–202.


Bertsekas, D. P. (1999).
Nonlinear programming. athena scientific belmont.
Massachusets, USA.


Boyd, S., Parikh, N., Chu, E., Peleato, B., Eckstein, J., et al. (2011).
Distributed optimization and statistical learning via the alternating direction method of multipliers.
Foundations and Trends® in Machine learning, 3(1):1–122.

References II


Daubechies, I., Defrise, M., and De Mol, C. (2004).
An iterative thresholding algorithm for linear inverse problems with a sparsity constraint.
Communications on Pure and Applied Mathematics: A Journal Issued by the Courant Institute of Mathematical Sciences, 57(11):1413–1457.


Donoho, D. L. (2006).
For most large underdetermined systems of linear equations the minimal l1-norm solution is also the sparsest solution.
Communications on Pure and Applied Mathematics: A Journal Issued by the Courant Institute of Mathematical Sciences, 59(6):797–829.


Elad, M. and Aharon, M. (2006).
Image denoising via sparse and redundant representations over learned dictionaries.
IEEE Transactions on Image processing, 15(12):3736–3745.


Engan, K., Aase, S. O., and Hakon Husoy, J. (1999).
Method of optimal directions for frame design.
In *1999 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings. ICASSP99 (Cat. No.99CH36258)*, volume 5, pages 2443–2446 vol.5.


Figueiredo, M. A., Nowak, R. D., and Wright, S. J. (2007).
Gradient projection for sparse reconstruction: Application to compressed sensing and other inverse problems.
IEEE Journal of selected topics in signal processing, 1(4):586–597.

References III


Gregor, K. and LeCun, Y. (2010).
Learning fast approximations of sparse coding.
In *Proceedings of the 27th International Conference on International Conference on Machine Learning*, pages 399–406. Omnipress.


Hubel, D. H. and Wiesel, T. N. (1959).
Receptive fields of single neurones in the cat's striate cortex.
The Journal of Physiology, 148(3):574–591.


Johnson, D. S. and Garey, M. R. (1979).
Computers and intractability: A guide to the theory of NP-completeness, volume 1.
WH Freeman San Francisco.


Koh, K., Kim, S.-J., and Boyd, S. (2007).
An interior-point method for large-scale l1-regularized logistic regression.
Journal of Machine learning research, 8(Jul):1519–1555.


Lesage, S., Gribonval, R., Bimbot, F., and Benaroya, L. (2005).
Learning unions of orthonormal bases with thresholded singular value decomposition.
In *Proceedings.(ICASSP'05). IEEE International Conference on Acoustics, Speech, and Signal Processing, 2005.*, volume 5, pages v–293. IEEE.

References IV


Li, Y. and Osher, S. (2009).
Coordinate descent optimization for l1 minimization with application to compressed sensing; a greedy algorithm.
Inverse Problems and Imaging, 3(3):487–503.


Luh, K. and Vu, V. (2016).
Dictionary learning with few samples and matrix concentration.
IEEE Transactions on Information Theory, 62(3):1516–1527.


Mallat, S. G. and Zhang, Z. (1993).
Matching pursuits with time-frequency dictionaries.
IEEE Transactions on signal processing, 41(12):3397–3415.


Natarajan, B. K. (1995).
Sparse approximate solutions to linear systems.
SIAM journal on computing, 24(2):227–234.


Olshausen, B. and Field, D. (1996).
Emergence of simple-cell receptive field properties by learning a sparse code for natural images.
Nature, 381:607–609.

References V


Papyan, V., Romano, Y., Sulam, J., and Elad, M. (2017).
Convolutional dictionary learning via local processing.
In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5296–5304.


Pati, Y. C., Rezaiifar, R., and Krishnaprasad, P. S. (1993).
Orthogonal matching pursuit: recursive function approximation with applications to wavelet decomposition.
In *Proceedings of 27th Asilomar Conference on Signals, Systems and Computers*, pages 40–44 vol.1.


Rockafellar, R. T. (1970).
Convex analysis, volume 28.
Princeton university press.


Rubinstein, R., Zibulevsky, M., and Elad, M. (2009).
Double sparsity: Learning sparse dictionaries for sparse signal approximation.
IEEE Transactions on signal processing, 58(3):1553–1564.


Schnass, K. (2015).
Local identification of overcomplete dictionaries.
Journal of Machine Learning Research, 16:1211–1242.

References VI


Sulam, J., Aberdam, A., Beck, A., and Elad, M. (2019).
On multi-layer basis pursuit, efficient algorithms and convolutional neural networks.
IEEE transactions on pattern analysis and machine intelligence.


Sulam, J., Papyan, V., Romano, Y., and Elad, M. (2018).
Multilayer convolutional sparse modeling: Pursuit and dictionary learning.
IEEE Transactions on Signal Processing, 66(15):4090–4104.


Sun, J., Qu, Q., and Wright, J. (2016).
Complete dictionary recovery over the sphere i: Overview and the geometric picture.
IEEE Transactions on Information Theory, 63(2):853–884.


Wu, S. and Yu, B. (2017).
Local identifiability of l_1-minimization dictionary learning: a sufficient and almost necessary condition.
The Journal of Machine Learning Research, 18(1):6121–6176.


Xu, Z. and Sun, J. (2010).
Image inpainting by patch propagation using patch sparsity.
IEEE transactions on image processing, 19(5):1153–1165.


Yang, A. Y., Zhou, Z., Balasubramanian, A. G., Sastry, S. S., and Ma, Y. (2013).
Fast l_1 minimization algorithms for robust face recognition.
IEEE Transactions on Image Processing, 22(8):3234–3246.

References VII

Yang, J., Yu, K., Gong, Y., Huang, T. S., et al. (2009).
Linear spatial pyramid matching using sparse coding for image classification.
In *CVPR*, volume 1, page 6.

Zeiler, M. D., Krishnan, D., Taylor, G. W., and Fergus, R. (2010).
Deconvolutional networks.
In *Cvpr*, volume 10, page 7.