# Compressed Sensing Meets Machine Learning
# - Classification of Mixture Subspace Models via Sparse Representation

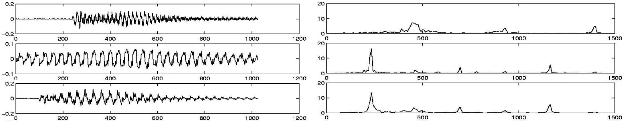
**Allen Y. Yang**
<yang@eecs.berkeley.edu>

Mini Lectures in Image Processing (Part I), UC Berkeley

# What is Sparsity

## Sparsity

A signal is sparse if most of its coefficients are (approximately) zero.

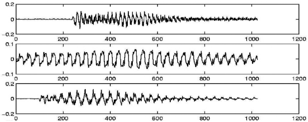



(a) Harmonic functions (b) Magnitude spectrum

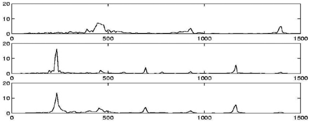
# What is Sparsity

## Sparsity

A signal is sparse if most of its coefficients are (approximately) zero.




(a) Harmonic functions




(b) Magnitude spectrum

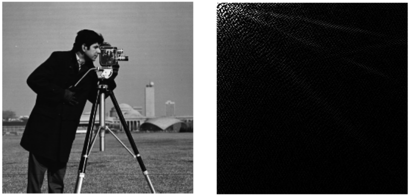
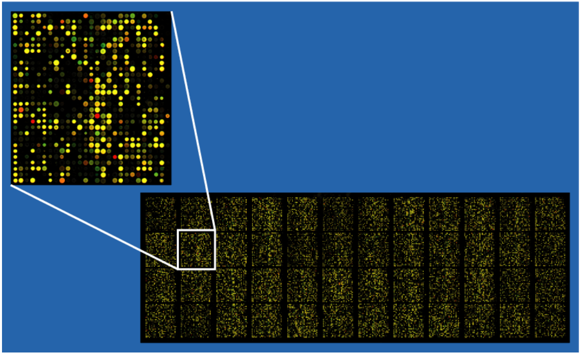


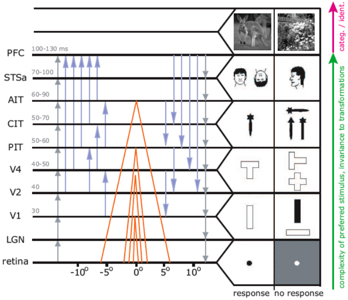
Figure: 2-D DCT transform.

# Sparsity in spatial domain

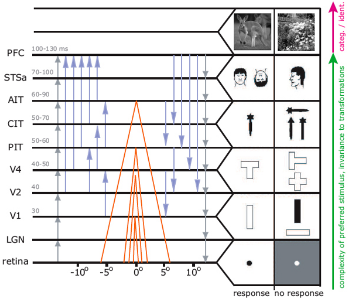
- gene microarray data [Drmanac et al. 1993]



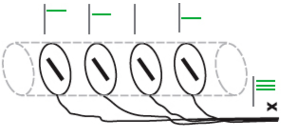
- **Sparsity in human visual cortex** [Olshausen & Field 1997, Serre & Poggio 2006]



- **Sparsity in human visual cortex** [Olshausen & Field 1997, Serre & Poggio 2006]




1. **Feed-forward**: No iterative feedback loop.
2. **Redundancy**: Average 80-200 neurons for each feature representation.
3. **Recognition**: Information exchange between stages is not about individual neurons, but rather how many neurons as a group fire together.



# Sparsity and $\ell^1$-Minimization

1. "Black gold" age [Claerbout & Muir 1973, Taylor, Banks & McCoy 1979]

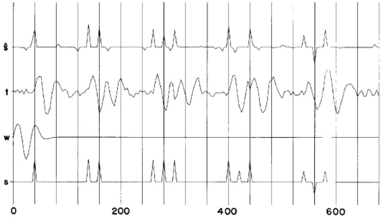



Figure: Deconvolution of spike train.

# Sparsity and $\ell^1$-Minimization

1. "Black gold" age [Claerbout & Muir 1973, Taylor, Banks & McCoy 1979]

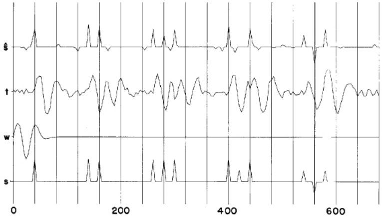



Figure: Deconvolution of spike train.

2. **Basis pursuit** [Chen & Donoho 1999]: Given $\mathbf{y} = A\mathbf{x}$ and $\mathbf{x}$ unknown,

$$\mathbf{x}^* = \arg\min \|\mathbf{x}\|_1, \text{ subject to } \mathbf{y} = A\mathbf{x}$$

3. **The Lasso** (least absolute shrinkage and selection operator) [Tibshirani 1996]

$$\mathbf{x}^* = \arg\min \|\mathbf{y} - A\mathbf{x}\|_2, \text{ subject to } \|\mathbf{x}\|_1 \leq k$$

# Taking Advantage of Sparsity

## What generates sparsity? (*d'après* Emmanuel Candès)

- Measure first, analyze later.
- Curse of dimensionality.

# Taking Advantage of Sparsity

**What generates sparsity? (*d'après* Emmanuel Candès)**

- Measure first, analyze later.
- Curse of dimensionality.

1. **Numerical analysis**: sparsity reduces cost for storage and computation.

# Taking Advantage of Sparsity

**What generates sparsity? (*d'après* Emmanuel Candès)**

- Measure first, analyze later.
- Curse of dimensionality.

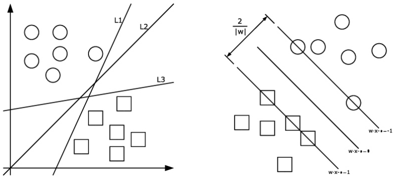
1. **Numerical analysis**: sparsity reduces cost for storage and computation.
2. **Compressed sensing** [Donoho 2006, Candes & Tao 2006]
   If signal $\mathbf{x}$ is sufficiently sparse, perfect reconstruction from $\mathbf{y} = A\mathbf{x}$ with sampling rate much lower than Shannon-Nyquist bound.

# Taking Advantage of Sparsity

**What generates sparsity? (*d'après* Emmanuel Candès)**

- Measure first, analyze later.
- Curse of dimensionality.

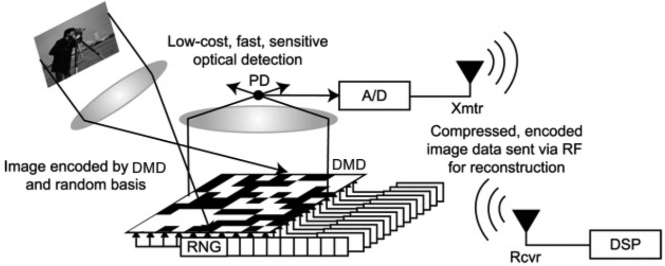
1. **Numerical analysis**: sparsity reduces cost for storage and computation.
2. **Compressed sensing** [Donoho 2006, Candes & Tao 2006]
   If signal $\mathbf{x}$ is sufficiently sparse, perfect reconstruction from $\mathbf{y} = A\mathbf{x}$ with sampling rate much lower than Shannon-Nyquist bound.
3. **Regularization in classification**:




(a) decision boundary (b) maximal margin

Figure: Linear support vector machine (SVM)

# One-Pixel Camera




$\mathbf{y} = A\mathbf{x}$, where $A \in \mathbb{R}^{d \times D}$ is a random projection matrix, $d \ll D$.

$$\begin{cases} y_1 & = & A(1,:)\mathbf{x} \\ y_2 & = & A(2,:)\mathbf{x} \\ \vdots & & \\ y_d & = & A(d,:)\mathbf{x} \end{cases}$$

# This Lecture

1. Classification via compressed sensing

2. Performance in face recognition

3. Extensions
   - Outlier rejection
   - Occlusion compensation

# Problem Formulation in Face Recognition

1. **Notations**
   - Training: For $K$ classes, collect training samples $\{\mathbf{v}_{1,1}, \cdots, \mathbf{v}_{1,n_1}\}, \cdots, \{\mathbf{v}_{K,1}, \cdots, \mathbf{v}_{K,n_K}\} \in \mathbb{R}^D$.
   - Test: Present a new $\mathbf{y} \in \mathbb{R}^D$, solve for label($\mathbf{y}$) $\in [1, 2, \cdots, K]$.

# Problem Formulation in Face Recognition

1. **Notations**
   - Training: For $K$ classes, collect training samples $\{\mathbf{v}_{1,1}, \cdots, \mathbf{v}_{1,n_1}\}, \cdots, \{\mathbf{v}_{K,1}, \cdots, \mathbf{v}_{K,n_K}\} \in \mathbb{R}^D$.
   - Test: Present a new $\mathbf{y} \in \mathbb{R}^D$, solve for label($\mathbf{y}$) $\in [1, 2, \cdots, K]$.
2. Construct $\mathbb{R}^D$ sample space via **stacking**

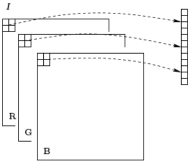



Figure: For images, assume 3-channel $640 \times 480$ image, $D = 3 \cdot 640 \cdot 480 \approx 1e6$.

# Problem Formulation in Face Recognition

1. **Notations**
   - Training: For $K$ classes, collect training samples $\{\mathbf{v}_{1,1}, \cdots, \mathbf{v}_{1,n_1}\}, \cdots, \{\mathbf{v}_{K,1}, \cdots, \mathbf{v}_{K,n_K}\} \in \mathbb{R}^D$.
   - Test: Present a new $\mathbf{y} \in \mathbb{R}^D$, solve for label$(\mathbf{y}) \in [1, 2, \cdots, K]$.
2. Construct $\mathbb{R}^D$ sample space via **stacking**

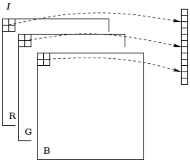
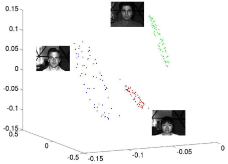



Figure: For images, assume 3-channel $640 \times 480$ image, $D = 3 \cdot 640 \cdot 480 \approx 1e6$.

3. Assume $\mathbf{y}$ belongs to Class $i$ [Belhumeur et al. 1997, Basri & Jacobs 2003]




$$
\begin{aligned}
\mathbf{y} &= \alpha_{i,1}\mathbf{v}_{i,1} + \alpha_{i,2}\mathbf{v}_{i,2} + \cdots + \alpha_{i,n_1}\mathbf{v}_{i,n_i}, \\
&= A_i \alpha_i,
\end{aligned}
$$

where $A_i = [\mathbf{v}_{i,1}, \mathbf{v}_{i,2}, \cdots, \mathbf{v}_{i,n_i}]$.

1. Nevertheless, $i$ is the variable we need to solve.
   Global representation:

$$\begin{aligned}\mathbf{y} &= [A_1, A_2, \cdots, A_K] \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_K \end{bmatrix}, \\ &= A\mathbf{x}_0.\end{aligned}$$

1. Nevertheless, $i$ is the variable we need to solve.
   Global representation:

$$\begin{aligned} \mathbf{y} &= [A_1, A_2, \cdots, A_K] \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_K \end{bmatrix}, \\ &= A\mathbf{x}_0. \end{aligned}$$

2. **Over-determined** system: $A \in \mathbb{R}^{D \times n}$, where $D \gg n = n_1 + \cdots + n_K$.
   $\mathbf{x}_0$ encodes membership of $\mathbf{y}$: If $\mathbf{y}$ belongs to Subject $i$,

$$\mathbf{x}_0 = [\, 0 \cdots 0 \;\alpha_i\; 0 \cdots 0 \,]^T \in \mathbb{R}^n.$$

1. Nevertheless, $i$ is the variable we need to solve.
   Global representation:

$$\begin{aligned}\mathbf{y} &= [A_1, A_2, \cdots, A_K]\begin{bmatrix}\alpha_1\\ \alpha_2\\ \vdots\\ \alpha_K\end{bmatrix},\\ &= A\mathbf{x}_0.\end{aligned}$$

2. **Over-determined** system: $A \in \mathbb{R}^{D\times n}$, where $D \gg n = n_1 + \cdots + n_K$.
   $\mathbf{x}_0$ encodes membership of $\mathbf{y}$: If $\mathbf{y}$ belongs to Subject $i$,

$$\mathbf{x}_0 = [\,0 \cdots 0\ \alpha_i\ 0 \cdots 0\,]^T \in \mathbb{R}^n.$$

### Problems to face

- Solving for $\mathbf{x}_0$ in $\mathbb{R}^D$ is **intractable**.
- True solution $\mathbf{x}_0$ is **sparse**: Average $\frac{1}{K}$ terms non-zero.

# Dimensionality Redunction

1. Construct linear projection $R \in \mathbb{R}^{d \times D}$, $d$ is the **feature dimension**, $d \ll D$.

$$\tilde{\mathbf{y}} \doteq R\mathbf{y} = RA\mathbf{x}_0 = \tilde{A}\mathbf{x}_0 \in \mathbb{R}^d.$$

$\tilde{A} \in \mathbb{R}^{d \times n}$, but $\mathbf{x}_0$ is unchanged.

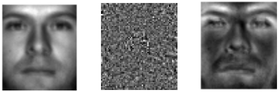
# Dimensionality Redunction

1. Construct linear projection $R \in \mathbb{R}^{d \times D}$, $d$ is the **feature dimension**, $d \ll D$.

$$\tilde{\mathbf{y}} \doteq R\mathbf{y} = RA\mathbf{x}_0 = \tilde{A}\mathbf{x}_0 \in \mathbb{R}^d.$$

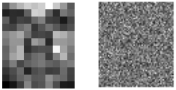
   $\tilde{A} \in \mathbb{R}^{d \times n}$, but $\mathbf{x}_0$ is unchanged.

2. Holistic features
   - Eigenfaces [Turk 1991]
   - Fisherfaces [Belhumeur 1997]
   - Laplacianfaces [He 2005]



3. Partial features

4. Unconventional features
   - Downsampled faces
   - Random projections



# Eigenfaces vs Fisherfaces

1. **Eigenfaces**: Principal component analysis (PCA)
   Denote projection vector $\mathbf{w} \in \mathbb{R}^D : \mathbf{w}^T \mathbf{x}_i = y_i \in \mathbb{R}$.

$$\mathbf{w}^* = \arg\max_{\mathbf{w}} \sum_{i=1}^{n} (y_i - \bar{y})^2 = \arg\max \mathbf{w}^T \Sigma \mathbf{w}.$$

# Eigenfaces vs Fisherfaces

1. **Eigenfaces**: Principal component analysis (PCA)
   Denote projection vector $\mathbf{w} \in \mathbb{R}^D : \mathbf{w}^T \mathbf{x}_i = y_i \in \mathbb{R}$.

$$\mathbf{w}^* = \arg\max_{\mathbf{w}} \sum_{i=1}^{n} (y_i - \bar{y})^2 = \arg\max \mathbf{w}^T \Sigma \mathbf{w}.$$

   **Numerical solution**: Singluar value decomposition (SVD)

$$\text{svd}(A) = U \Sigma V^T, \text{ where } U \in \mathbb{R}^{D\times D}, \Sigma \in \mathbb{R}^{D\times n}, V \in \mathbb{R}^{n\times n}.$$

   Denote $U = [U_1 \in \mathbb{R}^{D\times d}; U_2 \in \mathbb{R}^{D\times(D-d)}]$. Then $R = U_1^T$.

# Eigenfaces vs Fisherfaces

1. **Eigenfaces**: Principal component analysis (PCA)
   Denote projection vector $\mathbf{w} \in \mathbb{R}^D : \mathbf{w}^T \mathbf{x}_i = y_i \in \mathbb{R}$.

$$\mathbf{w}^* = \arg\max_{\mathbf{w}} \sum_{i=1}^{n} (y_i - \bar{y})^2 = \arg\max \mathbf{w}^T \Sigma \mathbf{w}.$$

   **Numerical solution**: Singluar value decomposition (SVD)

$$\mathrm{svd}(A) = U \Sigma V^T, \text{ where } U \in \mathbb{R}^{D \times D}, \Sigma \in \mathbb{R}^{D \times n}, V \in \mathbb{R}^{n \times n}.$$

   Denote $U = [U_1 \in \mathbb{R}^{D \times d}; U_2 \in \mathbb{R}^{D \times (D-d)}]$. Then $R = U_1^T$.

2. **Fisherfaces**: Linear discriminant analysis (LDA)
   Define **within class covariance matrix** $W = \Sigma_1 + \cdots + \Sigma_K$.

$$\mathbf{w}^* = \arg\max_{\mathbf{w}} \frac{\mathbf{w}^T \Sigma \mathbf{w}}{\mathbf{w}^T W \mathbf{w}}.$$

# Eigenfaces vs Fisherfaces

1. **Eigenfaces**: Principal component analysis (PCA)
   Denote projection vector $\mathbf{w} \in \mathbb{R}^D : \mathbf{w}^T \mathbf{x}_i = y_i \in \mathbb{R}$.

$$\mathbf{w}^* = \arg\max_{\mathbf{w}} \sum_{i=1}^{n} (y_i - \bar{y})^2 = \arg\max \mathbf{w}^T \Sigma \mathbf{w}.$$

   **Numerical solution**: Singluar value decomposition (SVD)

$$\mathrm{svd}(A) = U \Sigma V^T, \text{ where } U \in \mathbb{R}^{D\times D}, \Sigma \in \mathbb{R}^{D\times n}, V \in \mathbb{R}^{n\times n}.$$

   Denote $U = [U_1 \in \mathbb{R}^{D\times d}; U_2 \in \mathbb{R}^{D\times(D-d)}]$. Then $R = U_1^T$.

2. **Fisherfaces**: Linear discriminant analysis (LDA)
   Define **within class covariance matrix** $W = \Sigma_1 + \cdots + \Sigma_K$.

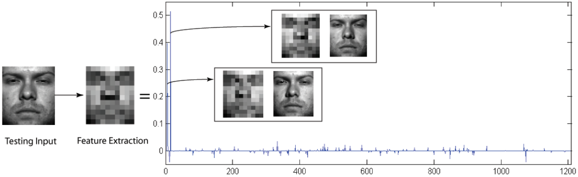
$$\mathbf{w}^* = \arg\max_{\mathbf{w}} \frac{\mathbf{w}^T \Sigma \mathbf{w}}{\mathbf{w}^T W \mathbf{w}}.$$

   **Numerical solution**: Generalized eigenvalue problem for $(\Sigma, W)$.

$$U = \mathrm{eig}(\Sigma, W).$$

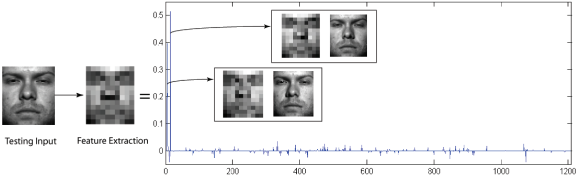
# $\ell^0$-Minimization

1. Ask for sparsest solution of underdetermined system $\tilde{\mathbf{y}} = \tilde{A}\mathbf{x}_0 \in \mathbb{R}^d$.



# $\ell^0$-Minimization

1. Ask for sparsest solution of underdetermined system $\tilde{\mathbf{y}} = \tilde{A}\mathbf{x}_0 \in \mathbb{R}^d$.



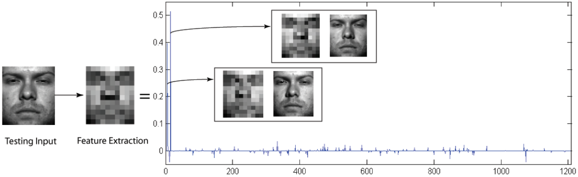

2. $\ell^0$-Minimization

$$\mathbf{x}_0 = \arg\min_{\mathbf{x}} \|\mathbf{x}\|_0 \ \text{s.t.}\ \tilde{\mathbf{y}} = \tilde{A}\mathbf{x}.$$

$\|\cdot\|_0$ simply counts the number of nonzero terms.

# $\ell^0$-Minimization

1. Ask for sparsest solution of underdetermined system $\tilde{\mathbf{y}} = \tilde{A}\mathbf{x}_0 \in \mathbb{R}^d$.



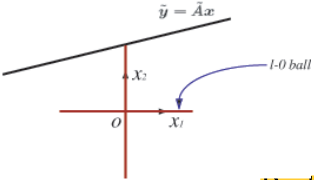

2. $\ell^0$-Minimization

$$\mathbf{x}_0 = \arg\min_{\mathbf{x}} \|\mathbf{x}\|_0 \text{ s.t. } \tilde{\mathbf{y}} = \tilde{A}\mathbf{x}.$$

$\|\cdot\|_0$ simply counts the number of nonzero terms.

3. $\ell^0$-Ball

- $\ell^0$-ball is not convex.
- $\ell^0$-minimization is NP-hard.



# $\ell^1/\ell^0$ Equivalence

1. **Compressed sensing**: If $\mathbf{x}_0$ is *sparse enough*, $\ell^0$-minimization is equivalent to

$$(P_1) \quad \min \|\mathbf{x}\|_1 \text{ s.t. } \tilde{\mathbf{y}} = \tilde{A}\mathbf{x}.$$

$\|\mathbf{x}\|_1 = |x_1| + |x_2| + \cdots + |x_n|.$

# $\ell^1/\ell^0$ Equivalence

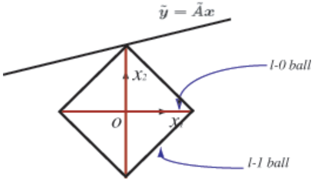
1. **Compressed sensing**: If $\mathbf{x}_0$ is *sparse enough*, $\ell^0$-minimization is equivalent to

$$(P_1) \quad \min \|\mathbf{x}\|_1 \text{ s.t. } \tilde{\mathbf{y}} = \tilde{A}\mathbf{x}.$$

$\|\mathbf{x}\|_1 = |x_1| + |x_2| + \cdots + |x_n|$.

2. $\ell^1$-Ball

- $\ell^1$-Minimization is convex.
- Solution equal to $\ell^0$-minimization.



# $\ell^1/\ell^0$ Equivalence

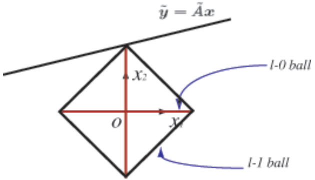
1. **Compressed sensing**: If $\mathbf{x}_0$ is *sparse enough*, $\ell^0$-minimization is equivalent to

$$(P_1) \quad \min \|\mathbf{x}\|_1 \text{ s.t. } \tilde{\mathbf{y}} = \tilde{A}\mathbf{x}.$$

$\|\mathbf{x}\|_1 = |x_1| + |x_2| + \cdots + |x_n|$.

2. $\ell^1$-Ball

- $\ell^1$-Minimization is convex.
- Solution equal to $\ell^0$-minimization.




3. $\ell^1/\ell^0$ Equivalence: [Donoho 2002, 2004; Candes et al. 2004; Baraniuk 2006]
Given $\tilde{\mathbf{y}} = \tilde{A}\mathbf{x}_0$, there exists **equivalence breakdown point** (EBP) $\rho(\tilde{A})$, if $\|\mathbf{x}_0\|_0 < \rho$:
   - $\ell^1$-solution is unique
   - $\mathbf{x}_1 = \mathbf{x}_0$

# $\ell^1$-Minimization Routines

- **Matching pursuit** [Mallat 1993]
  1. Find most correlated vector $\mathbf{v}_i$ in $\tilde{A}$ with $\mathbf{y}$: $i = \arg\max \langle \mathbf{y}, \mathbf{v}_j \rangle$.
  2. $\tilde{A} \leftarrow \tilde{A}^{\hat{i}}$, $x_i \leftarrow \langle \mathbf{y}, \mathbf{v}_i \rangle$, $\mathbf{y} \leftarrow \mathbf{y} - x_i \mathbf{v}_i$.
  3. Repeat until $\|\mathbf{y}\| < \epsilon$.
- **Basis pursuit** [Chen 1998]
  1. Assume $\mathbf{x}_0$ is $m$-sparse.
  2. Select $m$ linearly independent vectors $B_m$ in $\tilde{A}$ as a basis
  $$\mathbf{x}_m = B_m^\dagger \mathbf{y}.$$
  3. Repeat swapping one basis vector in $B_m$ with another vector in $\tilde{A}$ if improve $\|\mathbf{y} - B_m \mathbf{x}_m\|$.
  4. If $\|\mathbf{y} - B_m \mathbf{x}_m\|_2 < \epsilon$, stop.
- **Quadratic solvers**: $\tilde{\mathbf{y}} = \tilde{A}\mathbf{x}_0 + \mathbf{z} \in \mathbb{R}^d$, where $\|\mathbf{z}\|_2 < \epsilon$

$$\mathbf{x}^* = \arg\min\{\|\mathbf{x}\|_1 + \lambda \|\mathbf{y} - \tilde{A}\mathbf{x}\|_2\}$$

[Lasso, Second-order cone programming]: More expensive.

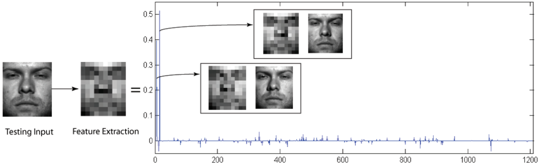
# $\ell^1$-Minimization Routines

- **Matching pursuit** [Mallat 1993]
  1. Find most correlated vector $\mathbf{v}_i$ in $\tilde{A}$ with $\mathbf{y}$: $i = \arg\max \langle \mathbf{y}, \mathbf{v}_j \rangle$.
  2. $\tilde{A} \leftarrow \tilde{A}^{\hat{i}}$, $x_i \leftarrow \langle \mathbf{y}, \mathbf{v}_i \rangle$, $\mathbf{y} \leftarrow \mathbf{y} - x_i \mathbf{v}_i$.
  3. Repeat until $\|\mathbf{y}\| < \epsilon$.
- **Basis pursuit** [Chen 1998]
  1. Assume $\mathbf{x}_0$ is $m$-sparse.
  2. Select $m$ linearly independent vectors $B_m$ in $\tilde{A}$ as a basis
  $$\mathbf{x}_m = B_m^{\dagger}\mathbf{y}.$$
  3. Repeat swapping one basis vector in $B_m$ with another vector in $\tilde{A}$ if improve $\|\mathbf{y} - B_m \mathbf{x}_m\|$.
  4. If $\|\mathbf{y} - B_m \mathbf{x}_m\|_2 < \epsilon$, stop.
- **Quadratic solvers**: $\tilde{\mathbf{y}} = \tilde{A}\mathbf{x}_0 + \mathbf{z} \in \mathbb{R}^d$, where $\|\mathbf{z}\|_2 < \epsilon$
$$\mathbf{x}^* = \arg\min\{\|\mathbf{x}\|_1 + \lambda \|\mathbf{y} - \tilde{A}\mathbf{x}\|_2\}$$
  [Lasso, Second-order cone programming]: More expensive.

## Matlab Toolboxes

- $\ell^1$-**Magic** by Candès at Caltech.
- **SparseLab** by Donoho at Stanford.
- **cvx** by Boyd at Stanford.

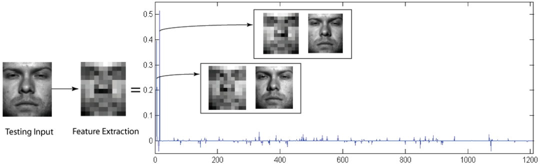
# Classification




1. Project $\mathbf{x}_1$ onto face subspaces:

$$\delta_1(\mathbf{x}_1) = \begin{bmatrix} \alpha_1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \delta_2(\mathbf{x}_1) = \begin{bmatrix} 0 \\ \alpha_2 \\ \vdots \\ 0 \end{bmatrix}, \cdots, \delta_K(\mathbf{x}_1) = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ \alpha_K \end{bmatrix}. \tag{1}$$

# Classification

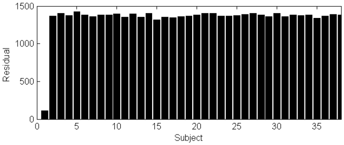



1. Project $\mathbf{x}_1$ onto face subspaces:

$$\delta_1(\mathbf{x}_1) = \begin{bmatrix} \alpha_1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \delta_2(\mathbf{x}_1) = \begin{bmatrix} 0 \\ \alpha_2 \\ \vdots \\ 0 \end{bmatrix}, \cdots, \delta_K(\mathbf{x}_1) = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ \alpha_K \end{bmatrix}. \tag{1}$$

2. Define residual $r_i = \|\tilde{\mathbf{y}} - \tilde{A}\delta_i(\mathbf{x}_1)\|_2$ for Subject $i$:

   - $\mathrm{id}(\mathbf{y}) = \arg\min_{i=1,\cdots,K}\{r_i\}$



# AR Database 100 Subjects (Illumination and Expression Variance)

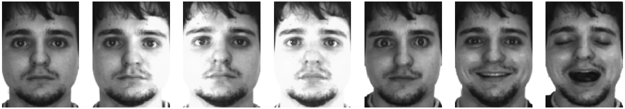


Table: I. Nearest Neighbor

| Dimension | 30 | 54 | 130 | 540 |
|---|---|---|---|---|
| Eigen [%] | 68.1 | 74.8 | 79.3 | 80.5 |
| Laplacian [%] | 73.1 | 77.1 | 83.8 | 89.7 |
| Random [%] | 56.7 | 63.7 | 71.4 | 75 |
| Down [%] | 51.7 | 60.9 | 69.2 | 73.7 |
| Fisher [%] | 83.4 | 86.8 | N/A | N/A |

Table: II. Nearest Subspace

| 30 | 54 | 130 | 540 |
|---|---|---|---|
| 64.1 | 77.1 | 82 | 85.1 |
| 66 | 77.5 | 84.3 | 90.3 |
| 59.2 | 68.2 | 80 | 83.3 |
| 56.2 | 67.7 | 77 | 82.1 |
| 80.3 | 85.8 | N/A | N/A |

Table: III. Linear SVM

| Dimension | 30 | 54 | 130 | 540 |
|---|---|---|---|---|
| Eigen [%] | 73 | 84.3 | 89 | 92 |
| Laplacian [%] | 73.4 | 85.8 | 90.8 | 95.7 |
| Random [%] | 54.1 | 70.8 | 81.6 | 88.8 |
| Down [%] | 51.4 | 73 | 83.4 | 90.3 |
| Fisher [%] | 86.3 | 93.3 | N/A | N/A |

Table: IV. $\ell^1$-Minimization

| 30 | 54 | 130 | 540 |
|---|---|---|---|
| 71.1 | 80 | 85.7 | 92 |
| 73.7 | 84.7 | 91 | 94.3 |
| 57.8 | 75.5 | 87.6 | 94.7 |
| 46.8 | 67 | 84.6 | 93.9 |
| 87 | 92.3 | N/A | N/A |

**Sparsity** vs. **Non-sparsity**: $\ell^1$ and SVM decisively outperform NN and NS.

1. Our framework seeks sparsity in representation of $\mathbf{y}$.
2. SVM seeks sparsity in decision boundaries on $A = [\mathbf{v}_1, \cdots, \mathbf{v}_n]$.
3. NN and NS do not enforce sparsity.

**Sparsity** vs. **Non-sparsity**: $\ell^1$ and SVM decisively outperform NN and NS.

1. Our framework seeks sparsity in representation of $\mathbf{y}$.
2. SVM seeks sparsity in decision boundaries on $A = [\mathbf{v}_1, \cdots, \mathbf{v}_n]$.
3. NN and NS do not enforce sparsity.

$\ell^1$-**Minimization** vs. **SVM**: Performance of SVM depends on the choice of features.

1. Random project performs poorly with SVMs.
2. $\ell^1$-Minimization guarantees performance convergence with different features.
3. At lower-dimensional space, *Fisher* features outperform.

Table: III. Linear SVM

| Dimension | 30 | 54 | 130 | 540 |
|---|---|---|---|---|
| Eigen [%] | 73 | 84.3 | 89 | 92 |
| Laplacian [%] | 73.4 | 85.8 | 90.8 | 95.7 |
| Random [%] | 54.1 | 70.8 | 81.6 | 88.8 |
| Down [%] | 51.4 | 73 | 83.4 | 90.3 |
| Fisher [%] | 86.3 | 93.3 | N/A | N/A |

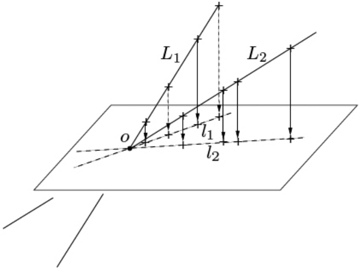
Table: IV. $\ell^1$-Minimization

| 30 | 54 | 130 | 540 |
|---|---|---|---|
| 71.1 | 80 | 85.7 | 92 |
| 73.7 | 84.7 | 91 | 94.3 |
| 57.8 | 75.5 | 87.6 | 94.7 |
| 46.8 | 67 | 84.6 | 93.9 |
| 87 | 92.3 | N/A | N/A |

# Randomfaces

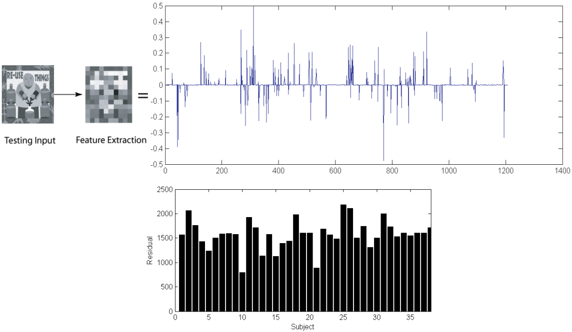
Blessing of Dimensionality [Donoho 2000]

- In high-dimensional data space $\mathbb{R}^D$, with **overwhelming probability**, $\ell^1/\ell^0$ equivalence holds for random projection $R$.
- **EBP**: $\rho \to 0.49d$ with both $n, d \to \infty$ proportionally!



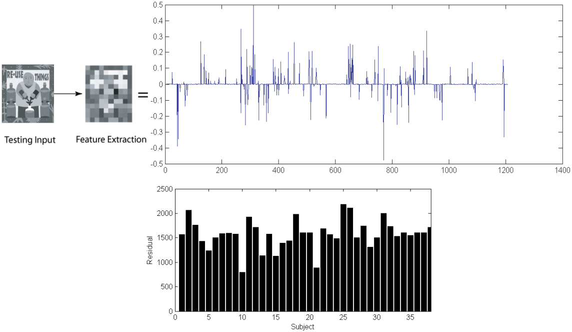
# Variation: Outlier Rejection

- $\ell^1$-Coefficients for invalid images



# Variation: Outlier Rejection

- $\ell^1$-Coefficients for invalid images

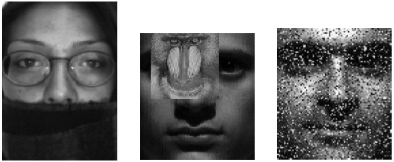



## Outlier Rejection

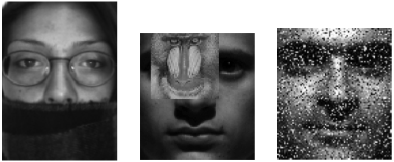
When $\ell^1$-solution is not sparse or concentrated to one subspace, the test sample is invalid.

$$\text{Sparsity Concentration Index: } \mathrm{SCI}(\mathbf{x}) \doteq \frac{K \cdot \max_i \|\delta_i(\mathbf{x})\|_1 / \|\mathbf{x}\|_1 - 1}{K-1} \in [0,1].$$

# Variation: Occlusion Compensation

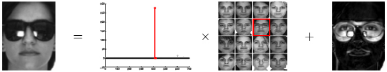


# Variation: Occlusion Compensation



1. Sparse representation + sparse error

$$\mathbf{y} = A\mathbf{x} + \mathbf{e}$$



2. Occlusion compensation:

$$\mathbf{y} = \begin{pmatrix} A & \mid & I \end{pmatrix} \begin{pmatrix} \mathbf{x} \\ \mathbf{e} \end{pmatrix} = B\mathbf{w}$$

**Reference**: *Robust face recognition via sparse representation*. Submitted to PAMI, 2008.

# Conclusion

1. **Sparsity** is important for classification of HD data.

# Conclusion

1. **Sparsity** is important for classification of HD data.
2. A new recognition framework via **compressed sensing**.

# Conclusion

1. **Sparsity** is important for classification of HD data.
2. A new recognition framework via **compressed sensing**.
3. In HD feature space, choosing an "optimal" feature becomes not significant.

# Conclusion

1. **Sparsity** is important for classification of HD data.
2. A new recognition framework via **compressed sensing**.
3. In HD feature space, choosing an "optimal" feature becomes not significant.
4. **Randomfaces**, **outliers**, **occlusion**.

# References


- **Donoho**. *For most large underdetermined systems of equations, the minimal $\ell^1$-norm near-solution approximates the sparsest near-solution*. 2004.
- **Candès**. *Compressive sampling*. 2006.
- **Donoho**. *Neighborly polytopes and sparse solution of underdetermined linear equations*. 2004.
- **Baraniuk et al**. *The Johnson-Lindenstrauss lemma meets compressed sensing*. 2006.