Medical Image Analysis and Processing

Image Noise Filtering

Sparse Denoising

Emad Fatemizadeh

Distance/online Course: Session 15

Date: 18 April 2021, 29th Farvardin 1400

Contents

Dictionary

- Dictionary Building:
- Concatenate fixed dictionary (Fourier, Dirac, DCT, Wavelet, etc.)

$$[D_1|D_2|\ldots|D_s] \in \mathbb{R}^{N\times(sN)}$$

- Concatenate samples from training data-set

$$[\boldsymbol{y}_1|\boldsymbol{y}_2|\ldots|\boldsymbol{y}_L] \in \mathbb{R}^{N\times L}$$

Dictionary

- Dictionary Learning: Find optimal representation (sparsest)! based on training samples
- Most popular approach (K-SVD)
- Consider matrix of training samples: $\mathbf{Y}$=$[\boldsymbol{y}_1|\boldsymbol{y}_2|\ldots|\boldsymbol{y}_L]$
- Corresponding sparse codes: $\mathbf{X}$=$[\boldsymbol{x}_1|\boldsymbol{x}_2|\ldots|\boldsymbol{x}_L]$

$$\min_{\mathbf{D},\mathbf{X}}\left\{\frac{1}{2}\|\mathbf{Y}-\mathbf{DX}\|_F^2\right\}, \qquad s.t. \forall i, \|\boldsymbol{x}_i\|_0 \leq T_0$$

- or alternatively:

$$\min_{\mathbf{D},\mathbf{X}}\left\{\sum_{i}\|\boldsymbol{x}_i\|_0\right\}, \qquad s.t. \|\mathbf{Y}-\mathbf{DX}\|_F^2 \leq \varepsilon$$

Sparse Image Denoising

- Sparse Image Denoising:
- Some Image Denoising algorithm sparse transform domain:
  - Naïve lowpass filter, sparsity frequency domain
  - Wavelet shrinkage, sparsify wavelet representation domain

Sparse Image Denoising

- Sparse Image Denoising {Naïve approach):
- $Y \in \mathbb{R}^n$: Noisy observation, vectorized patch of size $\sqrt{n} \times \sqrt{n}\{5 \le \sqrt{n} \le 20\}$
- $X \in \mathbb{R}^n$: Clean signal, vectorized patch of size $\sqrt{n} \times \sqrt{n}$
- $\mathrm{D} \in \mathbb{R}^{n \times k}$: A well-chosen overcomplete dictionary for our patch gallery! We assume each patch, *X*, has sparse representation with respect to *D*
- $\boldsymbol{\alpha} \in \mathbb{R}^k$: Representation in D, $\mathrm{X} = D\alpha$,
- How to denoise:

$$\hat{\alpha} = \underset{\alpha}{\operatorname{argmin}} \|\alpha\|_0, \qquad s.t. \|D\alpha - y\|_2^2 \le \varepsilon$$

$$\hat{X} = D\hat{\alpha}$$

Sparse Image Denoising

- Sparse Image Denoising (Naïve approach):
- We may solve this optimization problem:

$$\hat{\alpha} = \underset{\alpha}{\operatorname{argmin}}\{\|D\alpha - Y\|_2^2 + \lambda\|\alpha\|_0\}$$

- λ: Sparsity parameter

$$\hat{X} = D\hat{\alpha}$$

Sparse Image Denoising

› Blockwise Sparse Image Denoising:

› $\boldsymbol{X}$: Vectorized Clean Image (whole image, not patch)

› $\boldsymbol{Y}$: Vectorized Noisy Image (whole image, not patch)

$$\{\hat{\alpha}_{i,j}, \boldsymbol{X}\} = \underset{\{\alpha_{i,j}, \boldsymbol{X}\}}{\operatorname{argmin}} \left\{ \lambda \|\boldsymbol{X} - \boldsymbol{Y}\|_2^2 + \sum_{i,j} \left\| \boldsymbol{D}\alpha_{i,j} - \boldsymbol{R}_{i,j}\boldsymbol{X} \right\|_2^2 \right\}, \qquad s.t. \left\|\alpha_{i,j}\right\|_0 \le T_0$$

› Or equivalently:

$$\{\hat{\alpha}_{i,j}, \boldsymbol{X}\} = \underset{\{\alpha_{i,j}, \boldsymbol{X}\}}{\operatorname{argmin}} \left\{ \lambda \|\boldsymbol{X} - \boldsymbol{Y}\|_2^2 + \sum_{i,j} \left\| \boldsymbol{D}\alpha_{i,j} - \boldsymbol{R}_{i,j}\boldsymbol{X} \right\|_2^2 + \sum_{i,j} \mu_{i,j} \left\|\alpha_{i,j}\right\|_0 \right\}$$

Sparse Image Denoising

- Blockwise Sparse Image Denoising:

$$\{\hat{\alpha}_{i,j}, \boldsymbol{X}\} = \underset{\{\alpha_{i,j}, \boldsymbol{X}\}}{\operatorname{argmin}} \left\{ \lambda \|\boldsymbol{X} - \boldsymbol{Y}\|_2^2 + \sum_{i,j} \left\| \boldsymbol{D}\alpha_{i,j} - \boldsymbol{R}_{i,j}\boldsymbol{X} \right\|_2^2 + \sum_{i,j} \mu_{i,j} \left\| \alpha_{i,j} \right\|_0 \right\}$$

- The first term measure distance between the estimated and noisy images.
- The second (and third) terms make sure that each patch in the estimated image can be represented well as a linear combination of atoms from the dictionary D.

Sparse Image Denoising

- Blockwise Sparse Image Denoising:

$$\{\hat{\alpha}_{i,j}, \boldsymbol{X}\} = \underset{\{\alpha_{i,j}, \boldsymbol{X}\}}{\operatorname{argmin}} \left\{ \lambda \|\boldsymbol{X} - \boldsymbol{Y}\|_2^2 + \sum_{i,j} \left\| \boldsymbol{D}\alpha_{i,j} - \boldsymbol{R}_{i,j}\boldsymbol{X} \right\|_2^2 + \sum_{i,j} \mu_{i,j} \left\| \alpha_{i,j} \right\|_0 \right\}$$

- $\alpha_{i,j}$: Sparse representation of a $\sqrt{n} \times \sqrt{n}$ patch around pixel(i, j)
- $\boldsymbol{x}_{i,j} = \boldsymbol{R}_{i,j}\boldsymbol{X}$: $\sqrt{n} \times \sqrt{n}$ patch around pixel (i, j)
- $\boldsymbol{R}_{i,j} \in \mathbb{R}^{n \times N}$: Extracts the block patch (i, j) from the vectorized image.
- All image patches of size $\sqrt{n} \times \sqrt{n}$ (with overlaps) considered.

Sparse Image Denoising

- For learned (known) dictionary:
- Suppose $\boldsymbol{X} = \boldsymbol{Y}$, decouple and solve the following problem:

$$\hat{\alpha}_{i,j} = \underset{\alpha}{\operatorname{argmin}} \left\{ \left\| \boldsymbol{D}\alpha - \boldsymbol{x}_{i,j} \right\|_2^2 + \mu_{i,j} \|\alpha\|_0 \right\}$$

- Given all $\alpha_{i,j}$, decouple and solve the following problem:

$$\widehat{\boldsymbol{X}} = \underset{\boldsymbol{X}}{\operatorname{argmin}} \left\{ \lambda \|\boldsymbol{X} - \boldsymbol{Y}\|_2^2 + \sum_{i,j} \left\| \boldsymbol{D}\alpha_{i,j} - \boldsymbol{R}_{i,j}\boldsymbol{X} \right\|_2^2 \right\}$$

- Which has an explicit solution:

$$\widehat{\boldsymbol{X}} = \underset{\boldsymbol{X}}{\operatorname{argmin}} \left\{ \left(\lambda \boldsymbol{I} + \sum_{i,j} \boldsymbol{R}_{i,j}^T \boldsymbol{R}_{i,j} \right)^{-1} \left(\lambda \mathbf{Y} + \sum_{i,j} \boldsymbol{R}_{i,j}^T \boldsymbol{D} \hat{\alpha}_{i,j} \right) \right\}$$

- Repeat h times

Low Rank Representation and Denoising

› Basic idea:

Problem of low-rank matrix recovery and approximation

Singular Value Decomposition

- Let X be an $m \times l$ matrix and allow its rank, r, not to be necessarily full:

$$r \leq \min(m, l)$$

- Then there exist *unitary* matrices, $U \in \mathbb{R}^{m \times m}$ and $V \in \mathbb{R}^{l \times l}$, respectively, so that:

$$X = U\Sigma V^T = U \begin{bmatrix} D & O \\ O & O \end{bmatrix} V^T,$$

$$UU^T = I_{m \times m}, VV^T = I_{l \times l}$$

- Matrices denoted as O comprise zero elements and are of appropriate dimensions.

Singular Value Decomposition

- Singular Value Decomposition:

$$X = U\Sigma V^T = U \begin{bmatrix} D & 0 \\ 0 & 0 \end{bmatrix} V^T$$

- $D \in \mathbb{R}^{r\times r}, D = diag(\sigma_1, \sigma_2, \cdots, \sigma_r) = diag\left(\sqrt{\lambda_1}, \sqrt{\lambda_2} \cdots, \sqrt{\lambda_r}\right)$
- $\{\lambda_i\}_{i=1}^{r}$: are the *nonzero* eigenvalues of XX^T
- Let $\boldsymbol{u}_i$ and $\boldsymbol{v}_i$, denote the column vectors of matrices U and V:

$$U = [\boldsymbol{u}_1, \boldsymbol{u}_2, \cdots, \boldsymbol{u}_m]$$
$$V = [\boldsymbol{v}_1, \boldsymbol{v}_2, \cdots, \boldsymbol{v}_l]$$

Singular Value Decomposition

- As D is diagonal we can re-write:

$$X = [\boldsymbol{u}_1, \boldsymbol{u}_2, \cdots, \boldsymbol{u}_r] \begin{bmatrix} \sigma_1 & & & \\ & \sigma_2 & & \\ & & \ddots & \\ & & & \sigma_r \end{bmatrix} \begin{bmatrix} \boldsymbol{v}_1^T \\ \boldsymbol{v}_2^T \\ \vdots \\ \boldsymbol{v}_r^T \end{bmatrix}$$

- $\{\boldsymbol{u}_i\}_{i=1}^{r} \in \mathbb{R}^m$:Normalized eigenvectors corresponding to the nonzero eigenvalues of XX^T, $XX^T\boldsymbol{u}_i = \lambda_i \boldsymbol{u}_i$
- $\{\boldsymbol{v}_i\}_{i=1}^{r} \in \mathbb{R}^l$: Normalized eigenvectors corresponding to the nonzero eigenvalues of X^TX, $X^TX\boldsymbol{v}_i = \lambda_i \boldsymbol{v}_i$

Singular Value Decomposition

› An alternative for:

$$X = [\boldsymbol{u}_1, \boldsymbol{u}_2, \cdots, \boldsymbol{u}_r] \begin{bmatrix} \sigma_1 & & & \\ & \sigma_2 & & \\ & & \ddots & \\ & & & \sigma_r \end{bmatrix} \begin{bmatrix} \boldsymbol{v}_1^T \\ \boldsymbol{v}_2^T \\ \vdots \\ \boldsymbol{v}_r^T \end{bmatrix} = U_r D V_r^T,$$

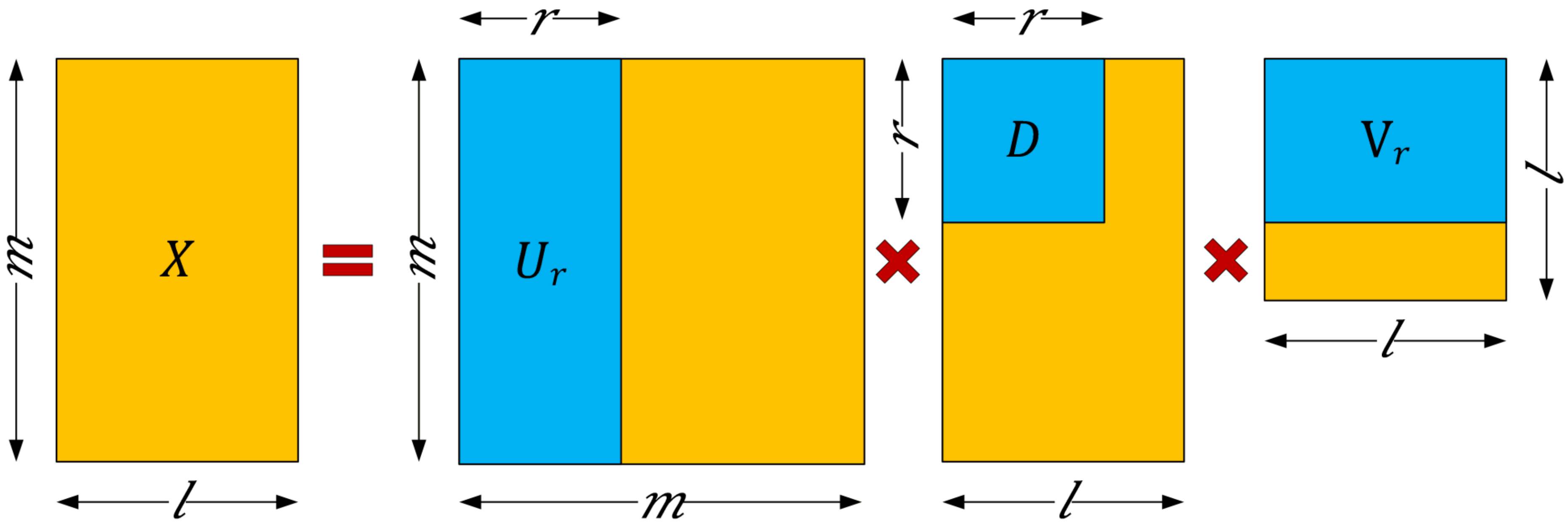
› where $U_r \in \mathbb{R}^{m\times r}$ $and\ U_r \in \mathbb{R}^{l\times r}$

› is:

$$X = \sum_{i=1}^{r} \sigma_i \boldsymbol{u}_i \boldsymbol{v}_i^T = \sum_{i=1}^{r} \sqrt{\lambda_i} \boldsymbol{u}_i \boldsymbol{v}_i^T$$

Singular Value Decomposition

› Graphical Illustration:



Singular Value Decomposition

› Example:

$$\begin{bmatrix} 1 & 2 & 3 \\ 3 & 2 & 1 \end{bmatrix} = \begin{bmatrix} -0.7071 & -0.7071 \\ -0.7071 & 0.7071 \end{bmatrix} \begin{bmatrix} 4.8990 & 0 & 0 \\ 0 & 2.000 & 0 \end{bmatrix} \begin{bmatrix} -0.5774 & 0.7071 & 0.4082 \\ -0.5774 & 0.0000 & -0.8165 \\ -0.5774 & -0.7071 & 0.4082 \end{bmatrix}^T$$

› or equivalently:

$$\begin{bmatrix} 1 & 2 & 3 \\ 3 & 2 & 1 \end{bmatrix} = \begin{bmatrix} -0.7071 & -0.7071 \\ -0.7071 & 0.7071 \end{bmatrix} \begin{bmatrix} 4.8990 & 0 \\ 0 & 2.000 \end{bmatrix} \begin{bmatrix} -0.5774 & 0.7071 \\ -0.5774 & 0.0000 \\ -0.5774 & -0.7071 \end{bmatrix}^T$$

Singular Value Decomposition

- Low Rank (k-Rank) approximation:
- Error free representation is:

$$X = \sum_{i=1}^{r} \sigma_i \boldsymbol{u}_i \boldsymbol{v}_i^T = \sum_{i=1}^{r} \sqrt{\lambda_i} \boldsymbol{u}_i \boldsymbol{v}_i^T$$

- A very interesting implication occurs if one uses less than r

$$X \simeq \hat{X} = \sum_{i=1}^{k} \sigma_i \boldsymbol{u}_i \boldsymbol{v}_i^T = \sum_{i=1}^{k} \sqrt{\lambda_i} \boldsymbol{u}_i \boldsymbol{v}_i^T, \qquad k < r$$

Singular Value Decomposition

- The best approximation error in Frobenius norm sense is given by:

$$\left\|X - \hat{X}\right\|_F^2 = \sum_{i=1}^{m}\sum_{j=1}^{l}\left|X(i,j) - \hat{X}(i,j)\right|^2 = \sum_{i=k+1}^{r} \sigma_i^2$$

- where:

$$\sigma_1^2 \geq \sigma_2^2 \geq \cdots \geq \sigma_r^2 > 0$$

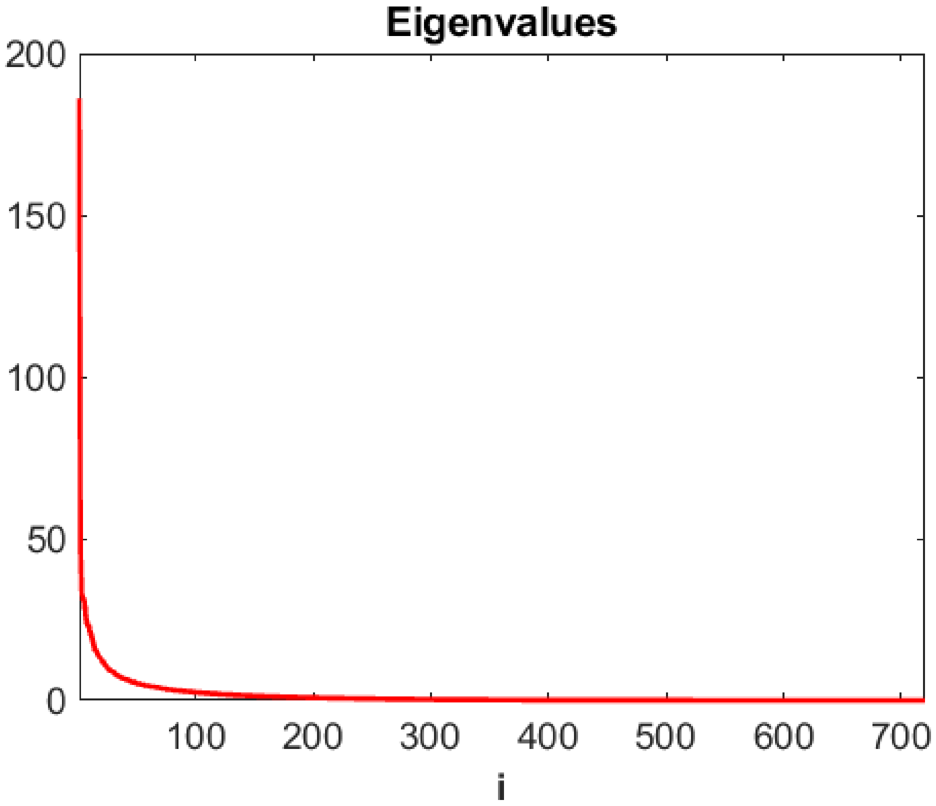
- Example:

$$\begin{bmatrix} 1 & 2 & 3 \\ 3 & 2 & 1 \end{bmatrix} \simeq \begin{bmatrix} 2 & 2 & 2 \\ 2 & 2 & 2 \end{bmatrix} = \begin{bmatrix} -0.7071 \\ -0.7071 \end{bmatrix} [4.8990] \begin{bmatrix} -0.5774 \\ -0.5774 \\ -0.5774 \end{bmatrix}^T$$

MEDICAL IMAGE ANALYSIS AND PROCESSING, E.FATEMIZADEH (2020)

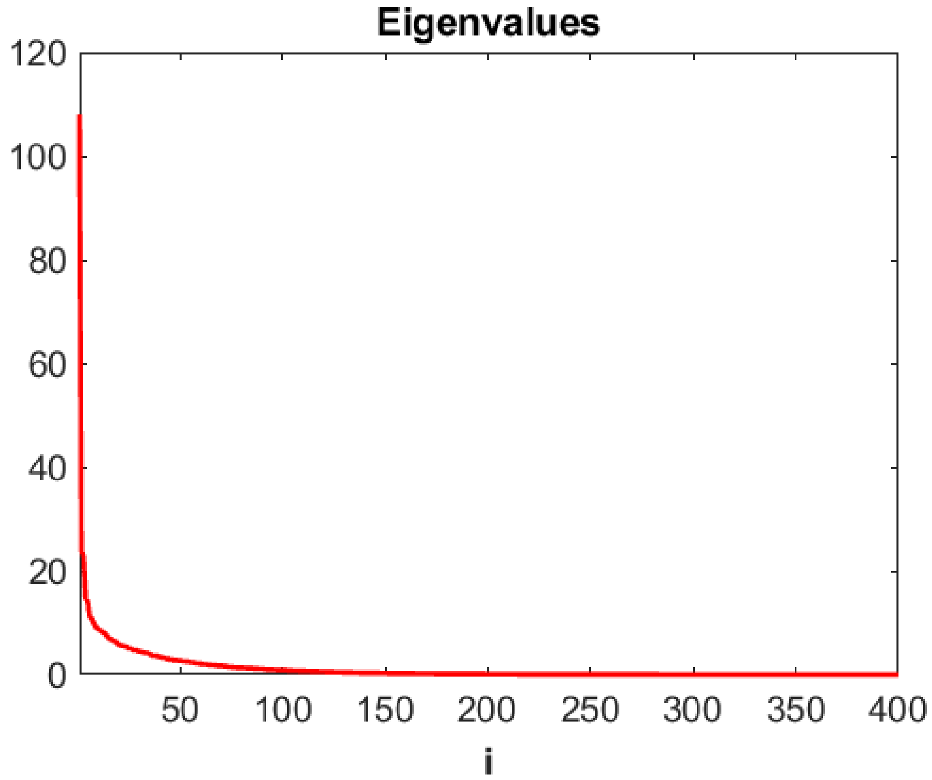
Singular Value Decomposition

› Some Medical Example: X=MRI images!



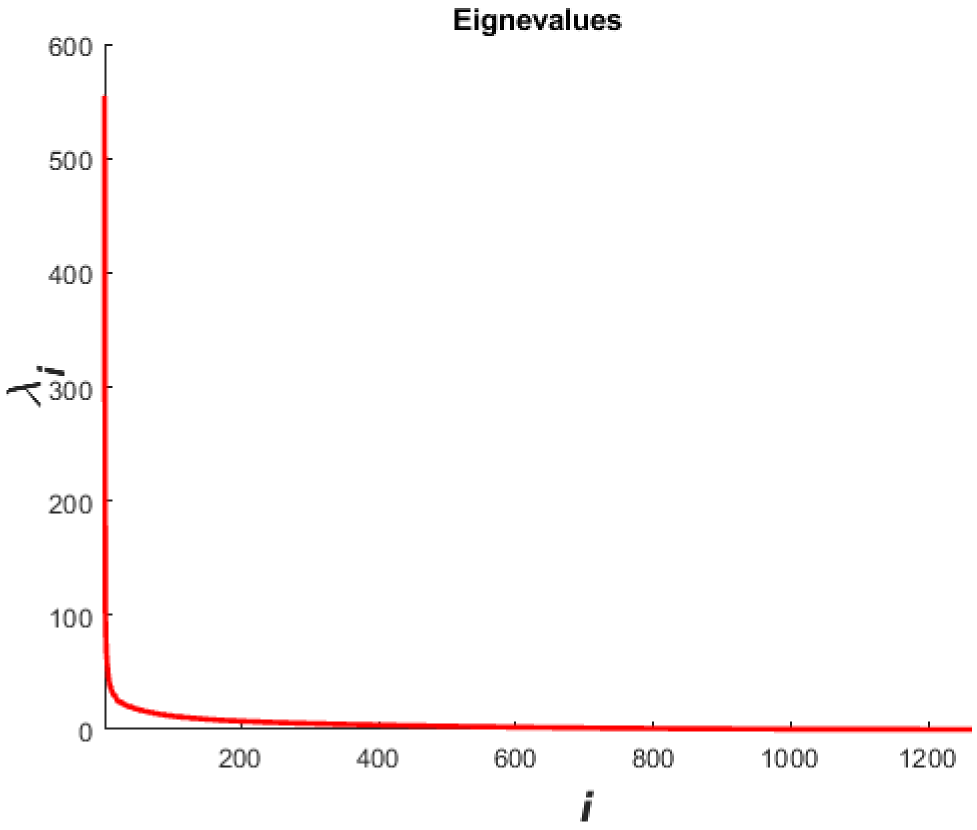
Singular Value Decomposition

› Some Medical Example: X=MRI images!



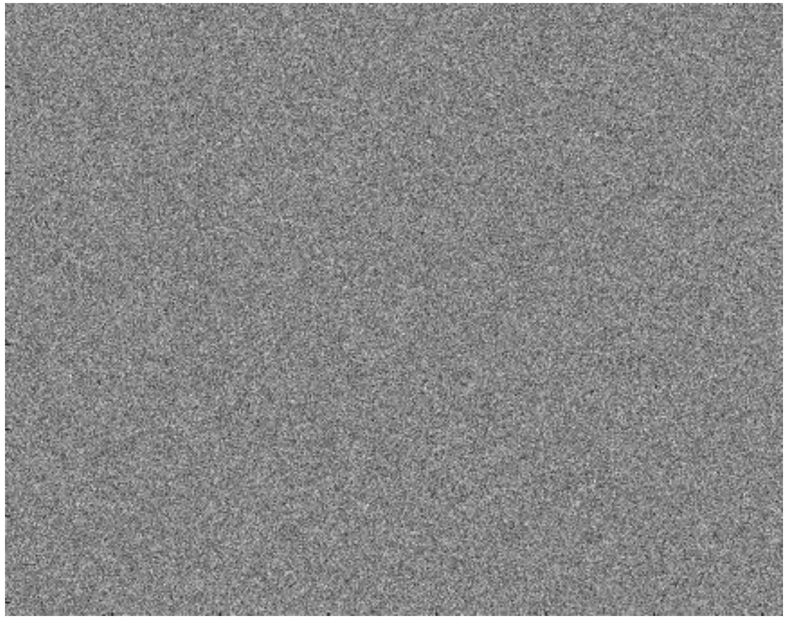
Singular Value Decomposition

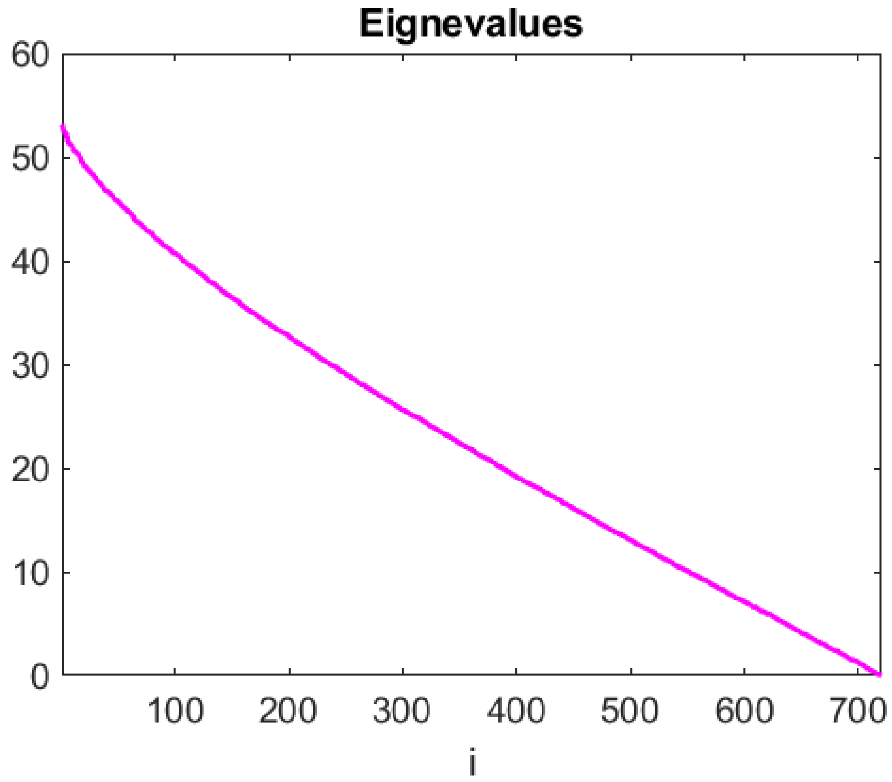
› Some Artificial Example: X=painting images!



Singular Value Decomposition

› Some Medical Example: X=noise!





Low-Rank Matric Recovery

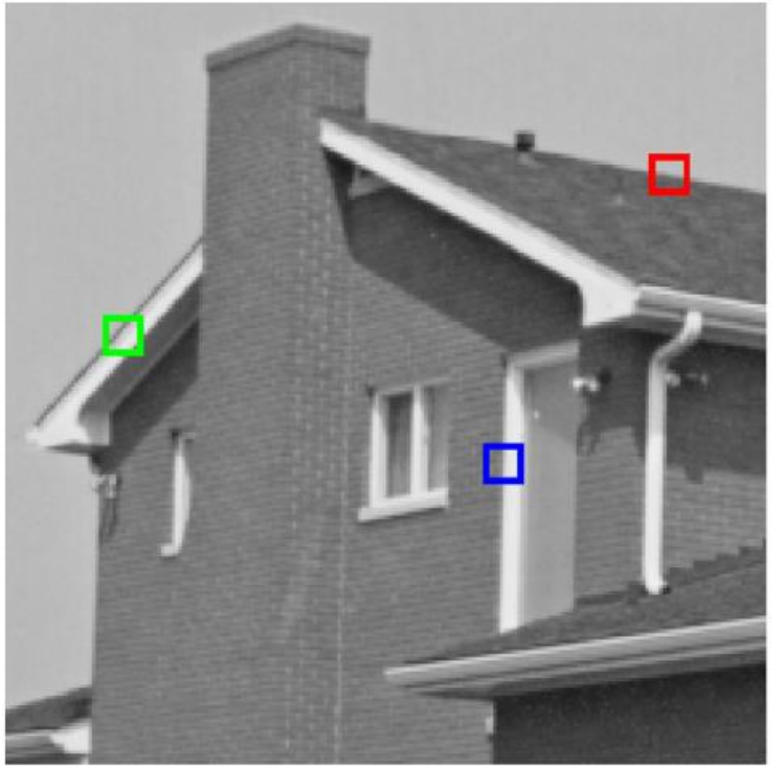
- Problem of Low-Rank Matric Recovery:
- Estimating the latent low-rank matrix $\boldsymbol{X}$ from its noisy observation $\boldsymbol{Y}$:

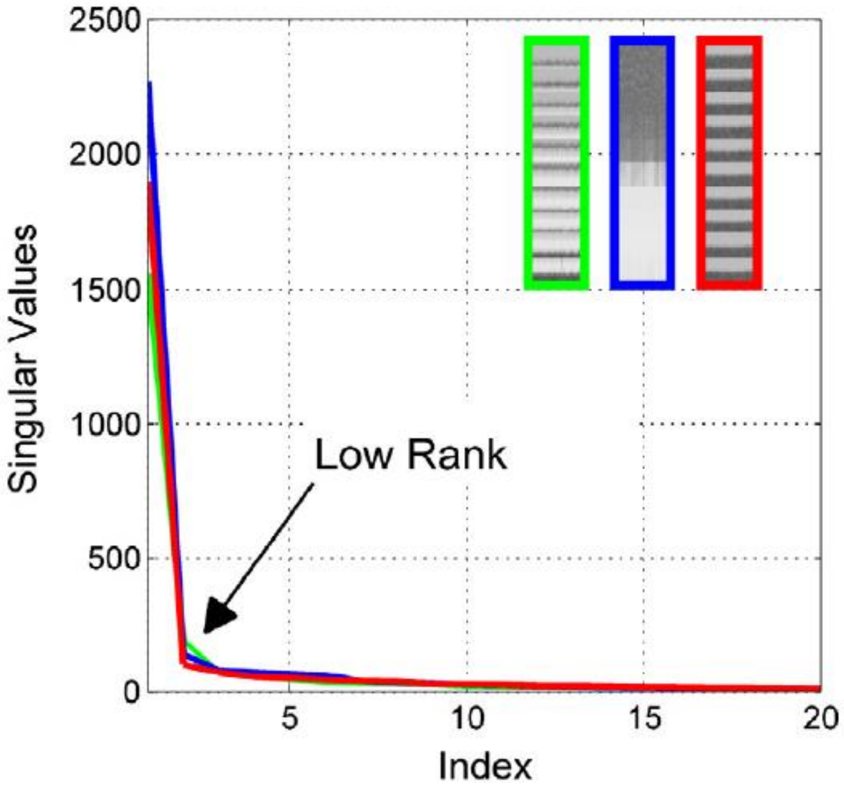
$$\boldsymbol{Y} = \boldsymbol{X} + \boldsymbol{\sigma W}$$

- Where $\boldsymbol{Y} = [\boldsymbol{y}_1, \boldsymbol{y}_2, \cdots, \boldsymbol{y}_m] \in \mathbb{R}^{m\times n}$ is patch matrix composed by m similar $\sqrt{n} \times \sqrt{n}$ vectorized patch ($\boldsymbol{y}_i \in \mathbb{R}^n$) from noisy image and $\boldsymbol{W}$ the noise matrix with $i.i.d.$ entries, $W_{i,j} \sim N(0,1)$.
- Why similar patch? Hint, what is rank of $\boldsymbol{Y} = [\boldsymbol{y}_1, \boldsymbol{y}_1, \cdots, \boldsymbol{y}_1]$?

Low-Rank Matric Recovery

› Example:





Low-Rank Matric Recovery

- A naïve and natural approach for approximating a low-rank matrix from noisy data consists in truncating, or hard-thresholding of the singular values of the observed matrix $\boldsymbol{Y}$.

$$\boldsymbol{X} \simeq \sum_{i=1}^{r} \sigma_i h(\sigma_i - \theta)\boldsymbol{u}_i\boldsymbol{v}_i^T$$

- where h is Heaviside (step) function and θ is threshold parameter.
- An alternative estimator is soft-thresholding

$$\boldsymbol{X} \simeq \sum_{i=1}^{r} max(\sigma_i - \theta, 0)\boldsymbol{u}_i\boldsymbol{v}_i^T$$

Low-Rank Matric Recovery

- Most popular approach is low rank minimization problem

$$\widehat{\boldsymbol{X}} = \min_{X} \left\{ \frac{1}{2} \|\boldsymbol{X} - \boldsymbol{Y}\|_F^2 + \lambda \|\boldsymbol{X}\|_* \right\}$$

- Where $\|\boldsymbol{X}\|_*$ is nuclear norm:

$$\|\boldsymbol{X}\|_* = \sum_{i=1}^{r} \sigma_i\left(\boldsymbol{X}\right)$$

- It is shown that the basic solution is:
- $\boldsymbol{X} = \boldsymbol{U}\boldsymbol{\Sigma}_{\lambda}\boldsymbol{V}^{\boldsymbol{T}}$, where $\mathbf{Y} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^{\boldsymbol{T}}$(SVD of noisy observation) and:

$$\boldsymbol{\Sigma}_{\lambda} = \max\{\boldsymbol{\Sigma} - \lambda\boldsymbol{I}, \boldsymbol{0}\}$$

The End

› AnY QuEsTiOn?