Singular Value Decomposition - Applications in Image Processing

Iveta Hnětynková

Katedra numerické matematiky, MFF UK

Ústav informatiky, AV ČR

Outline

1. Singular value decomposition
2. Application 1 - image compression
3. Application 2 - image deblurring

1. Singular value decomposition

Consider a (real) matrix

$$A \in \mathcal{R}^{n \times m}, \;\; r = \text{rank}\,(A) \leq \min\,\{n, m\}\,.$$

A has

m	columns of length n ,
n	rows of lenght m ,
r	is the maximal number of linearly independent columns (rows) of A .

There exists an **SVD decomposition** of A in the form

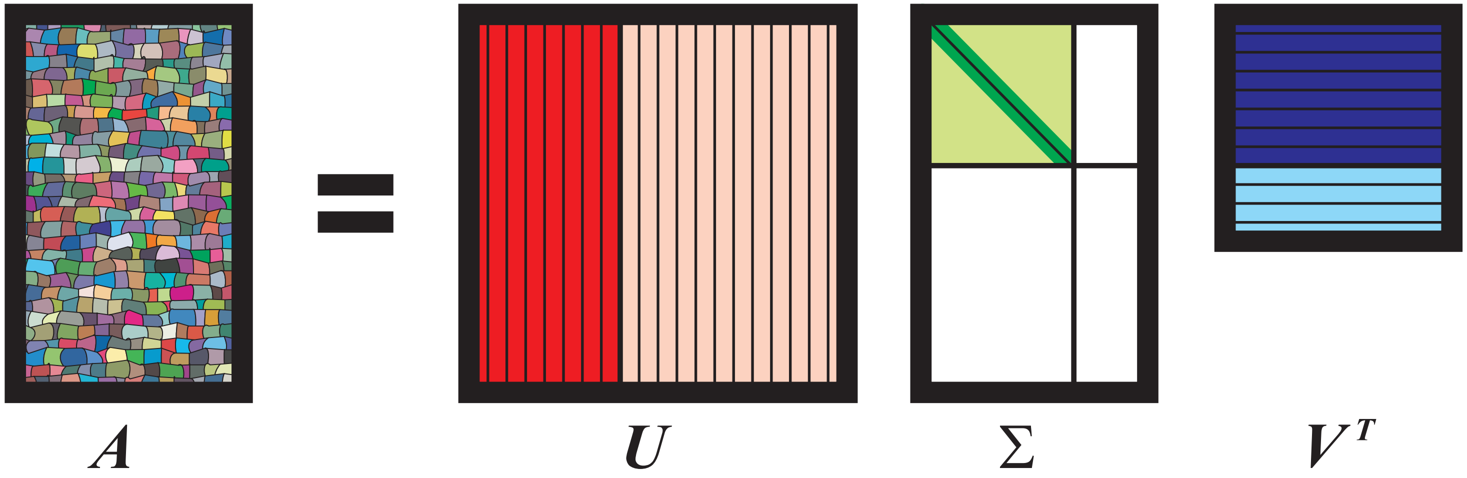
$$A = U \Sigma V^T,$$

where $U = [u_1, \ldots, u_n] \in \mathcal{R}^{n \times n}, V = [v_1, \cdots, v_m] \in \mathcal{R}^{m \times m}$ are orthogonal matrices, and

$$\Sigma = \begin{bmatrix} \Sigma_r & 0 \\ 0 & 0 \end{bmatrix} \in \mathcal{R}^{n \times m}, \quad \Sigma_r = \begin{bmatrix} \sigma_1 & & \\ & \ddots & \\ & & \sigma_r \end{bmatrix} \in \mathcal{R}^{r \times r},$$

$$\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_r > 0\,.$$

Singular value decomposition – the matrices:



$\{u_i\}_{i=1,\ldots,n}$ are **left singular vectors** (columns of U),

$\{v_i\}_{i=1,\ldots,m}$ are **right singular vectors** (columns of V),

$\{\sigma_i\}_{i=1,\ldots,r}$ are **singular values** of A.

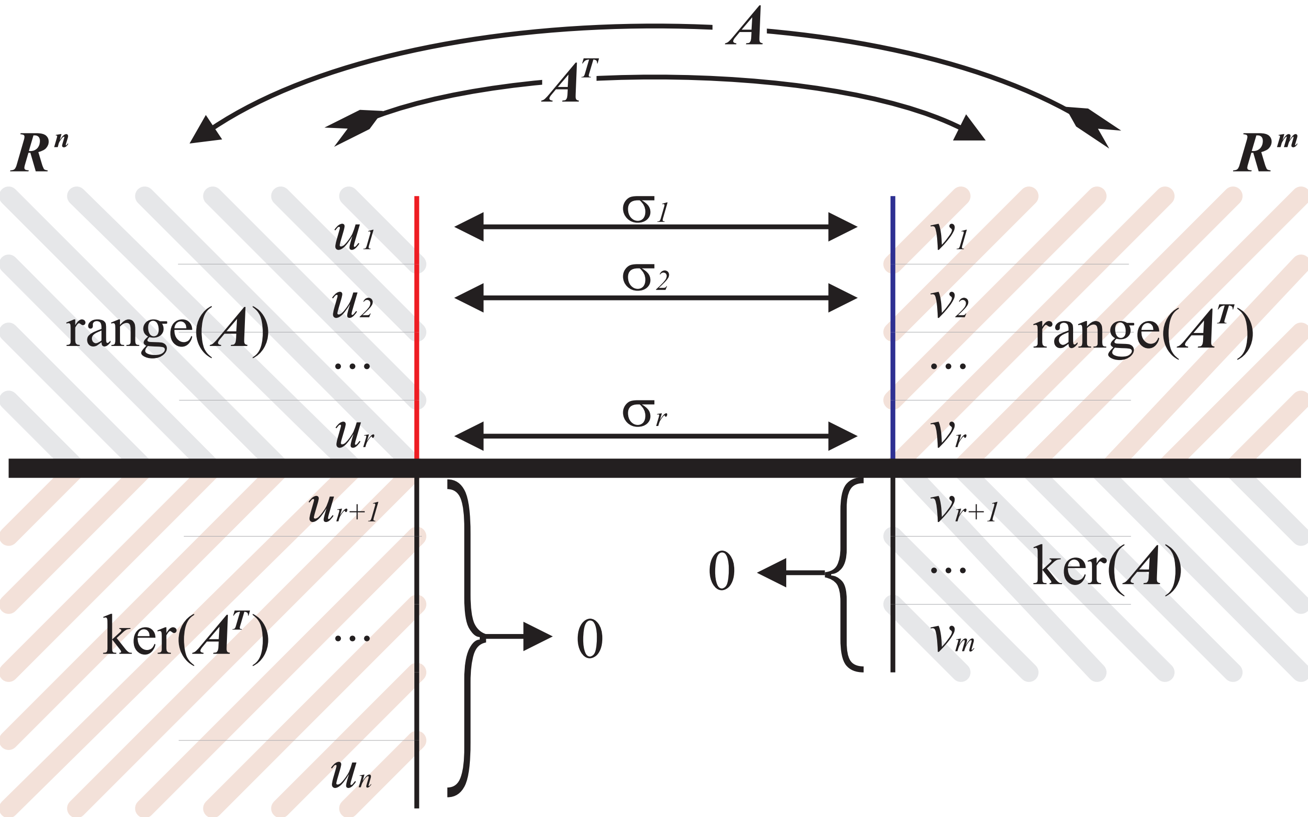
The SVD gives us:

$$\begin{aligned} \text{span}\,(u_1,\ldots,u_r) &\equiv \text{range}\,(\text{A}) \subset \mathcal{R}^n, \\ \text{span}\,(v_{r+1},\ldots,v_m) &\equiv \text{ker}\,(\text{A}) \subset \mathcal{R}^m, \end{aligned}$$

$$\begin{aligned} \text{span}\,(v_1,\ldots,v_r) &\equiv \text{range}\,(\text{A}^\text{T}) \subset \mathcal{R}^m, \\ \text{span}\,(u_{r+1},\ldots,u_n) &\equiv \text{ker}\,(\text{A}^\text{T}) \subset \mathcal{R}^n, \end{aligned}$$

spectral and Frobenius norm of A, rank of A, ...

Singular value decomposition – the subspaces:



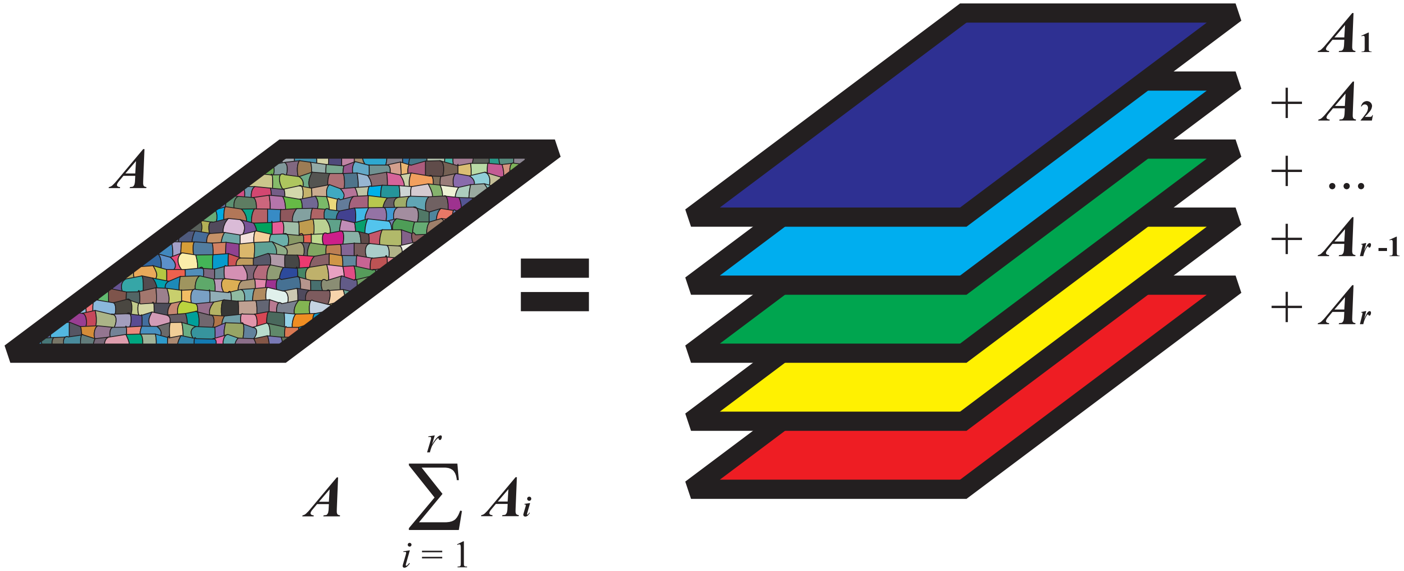
The outer product (dyadic) form:

We can rewrite A as a sum of rank-one matrices in the dyadic form

$$
\begin{aligned}
A &= U \Sigma V^T \\
&= [u_1, \ldots, u_r] \begin{bmatrix} \sigma_1 & & \\ & \ddots & \\ & & \sigma_r \end{bmatrix} \begin{bmatrix} v_1^T \\ \vdots \\ v_r^T \end{bmatrix} \\
&= u_1 \sigma_1 v_1^T + \ldots + u_r \sigma_r v_r^T \\
&= \sum_{i=1}^{r} \sigma_i u_i v_i^T \\
&\equiv \sum_{i=1}^{r} A_i \, .
\end{aligned}
$$

Moreover $\|A_i\|_2 = \sigma_i$ gives $\|A_1\|_2 \geq \|A_2\|_2 \geq \ldots \geq \|A_r\|_2$.

Matrix A as a sum of rank-one matrices:



SVD reveals the dominating information encoded in a matrix. The first terms are the "most" important.

Optimal approximation of A with a rank-k:

The sum of the first k dyadic terms

$$\sum_{i=1}^{k} A_i \equiv \sum_{i=1}^{k} \sigma_i u_i v_i^T$$

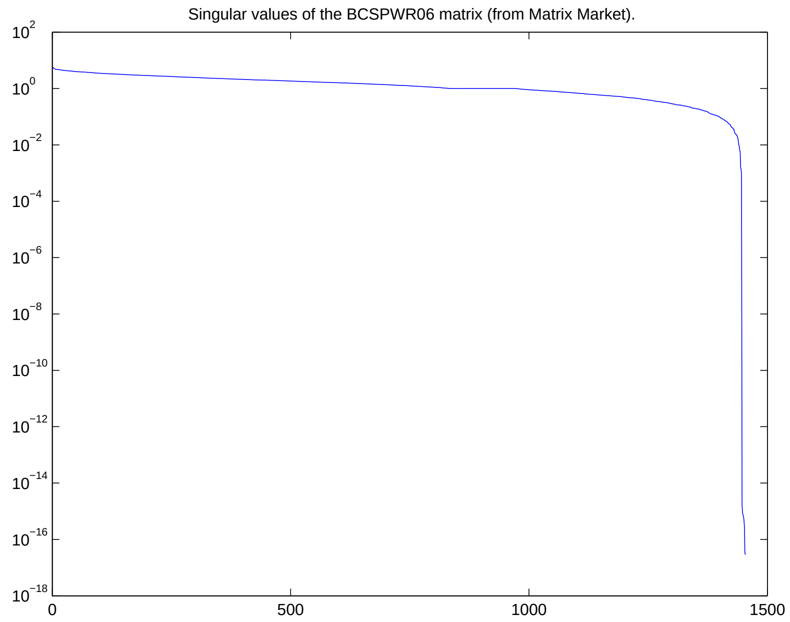
is the best rank-k approximation of the matrix A in the sense of minimizing the 2-norm of the approximation error, tj.

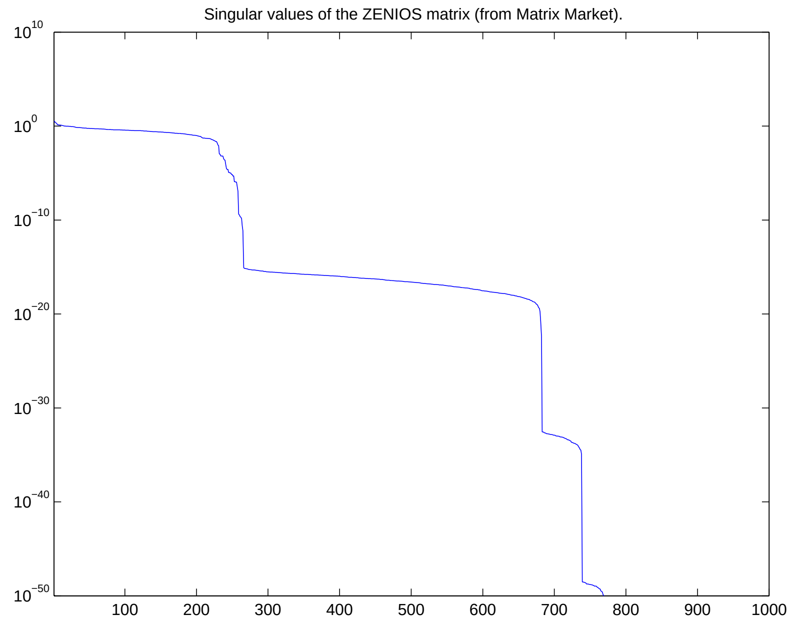
$$\sum_{i=1}^{k} u_i \sigma_i v_i^T = \arg \min_{X \in \mathcal{R}^{n\times m},\, \mathrm{rank}\,(X) \le k} \{\|A - X\|_2\}$$

This allows to approximate A with a lower-rank matrix

$$A \approx \sum_{i=1}^{k} A_i \equiv \sum_{i=1}^{k} \sigma_i u_i v_i^T \,.$$

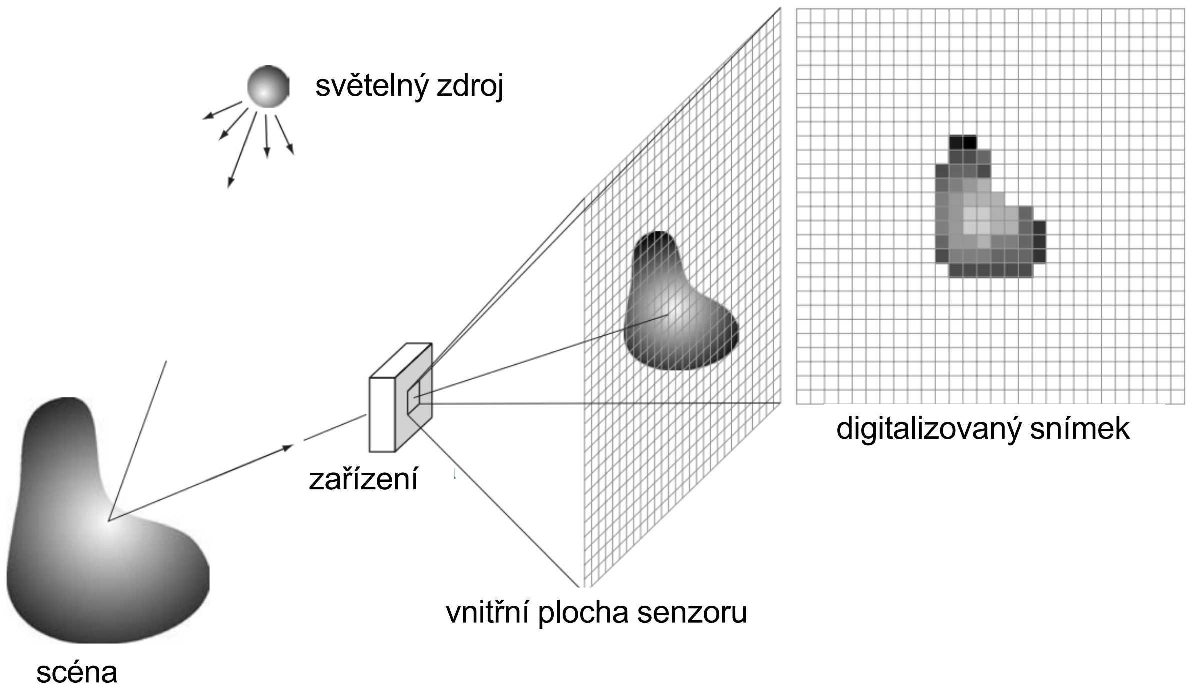
Different possible distributions of singular values:





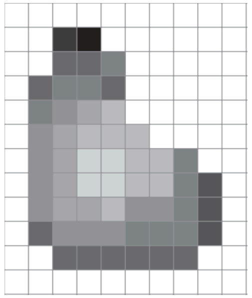
The hardly (left) and the easily (right) approximable matrices (BCSPWR06 and ZENIOS from the Harwell-Boeing Collection).

2. Application 1 - image compression



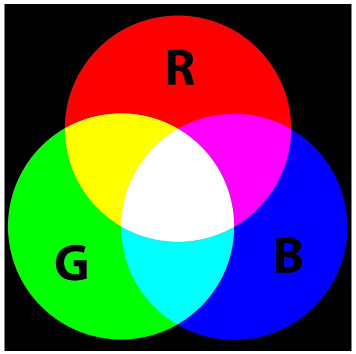
Grayscale image = matrix, each entry represents a pixel brightness.

Grayscale image: scale $0, \ldots, 255$ from black to white



$$
= \begin{bmatrix}
255 & 255 & 255 & 255 & 255 & \ldots & 255 & 255 & 255 \\
255 & 255 & 31 & 0 & 255 & \ldots & 255 & 255 & 255 \\
255 & 255 & 101 & 96 & 121 & \ldots & 255 & 255 & 255 \\
255 & 99 & 128 & 128 & 98 & \ldots & 255 & 255 & 255 \\
\vdots & \vdots & \vdots & \vdots & \vdots & \ldots & \vdots & \vdots & \vdots \\
255 & 90 & 158 & 153 & 158 & \ldots & 100 & 35 & 255 \\
255 & 255 & 102 & 103 & 99 & \ldots & 98 & 255 & 255 \\
255 & 255 & 255 & 255 & 255 & \ldots & 255 & 255 & 255
\end{bmatrix}
$$

Colored image: 3 matrices for Red, Green and Blue brightness values



MATLAB DEMO:

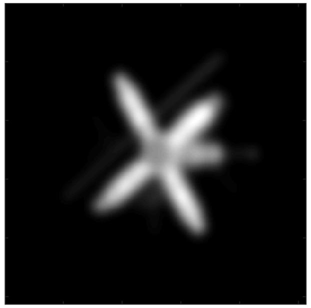
Approximate a grayscale image A using the SVD by $\sum_{i=1}^{k} A_i$. Compare storage requirements and quality of approximation for different k.

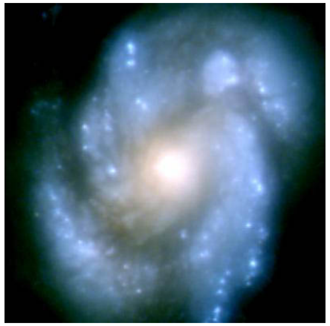
Memory required to store:

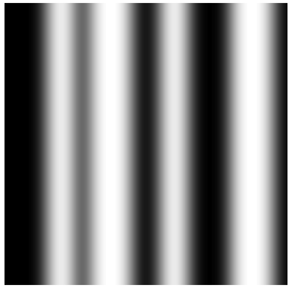
an uncompressed image of size $m \times n$: mn values

rank k SVD approximation: $k(m+n+1)$ values

3. Application 2 - image deblurring







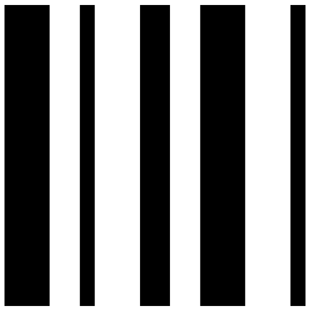
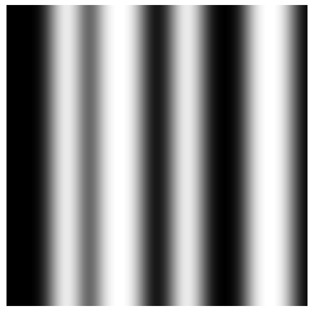
Sources of noise and blurring:

- physical sources (moving objects, lens out of focus, ...),
- measurement errors,
- discretization errors,
- rounding errors (finite precision arithmetics in computer),
- ...

Challenge: Given only the blurred noisy image B and having information about how it was obtained try to find (or approximate) exact image X.

Model of blurring process:

Blurred photo:  $\longrightarrow$ 

Barcode scanning:  $\longrightarrow$ 

X(exact image) $\mathcal{A}$(blurring operator) B(blurred noisy image)

Obtaining a linear model:

Using some discretization techniques, it is possible to transform this problem to a linear problem

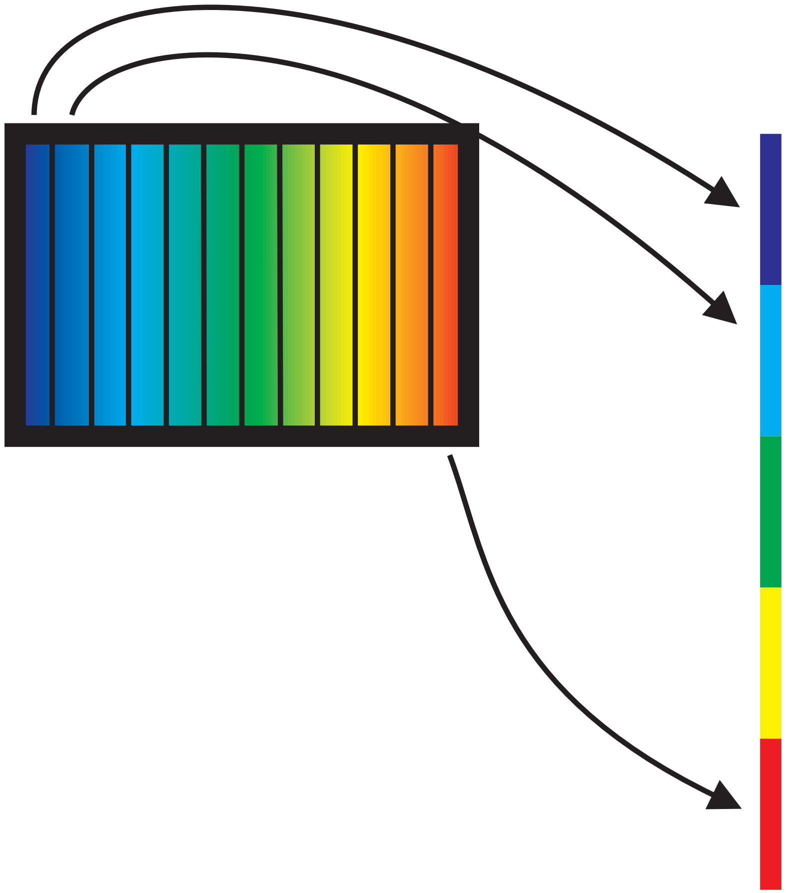
$$Ax = b, \quad A \in \mathcal{R}^{n \times n}, \quad x, b \in \mathcal{R}^n,$$

where A is a discretization of $\mathcal{A}$, $b = \text{vec}(B)$, $x = \text{vec}(X)$.

Size of the problem: $n =$ number of pixels in the image, e.g., even for a low resolution 456 x 684 px we get 723 432 equations.

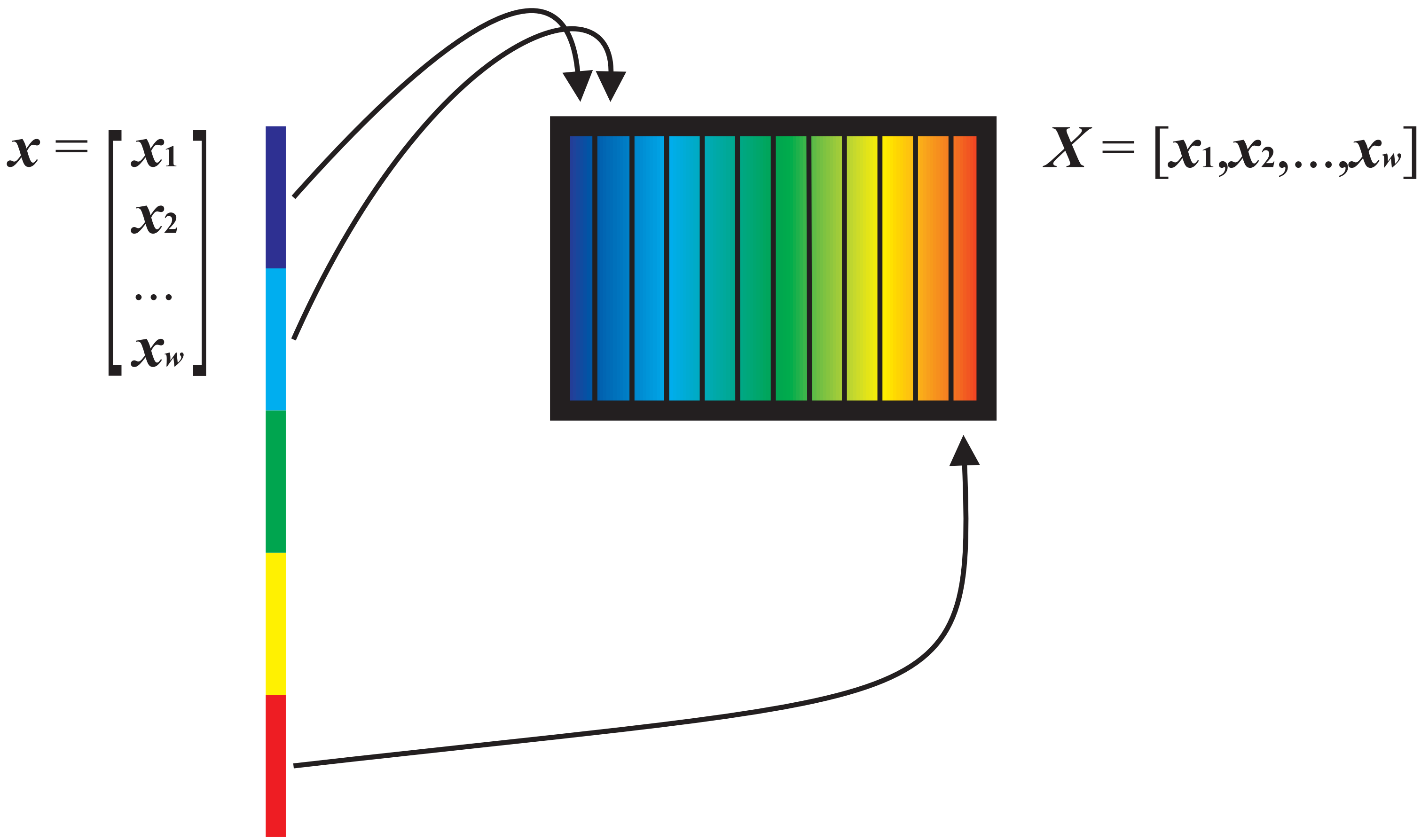
Image vectorization $B \rightarrow b = \mathrm{vec}\,(B)$**:**

$$\boldsymbol{B} = [\boldsymbol{b}_1, \boldsymbol{b}_2, \ldots, \boldsymbol{b}_w]$$



$$\boldsymbol{b} = \begin{bmatrix} \boldsymbol{b}_1 \\ \boldsymbol{b}_2 \\ \ldots \\ \boldsymbol{b}_w \end{bmatrix}$$

Solution back reshaping $x = \text{vec}(X) \rightarrow X$:



Solution of the linear problem:

Let A be nonsingular (which is usually the case). Then

$$Ax = b$$

has the unique solution

$$x = A^{-1}b.$$

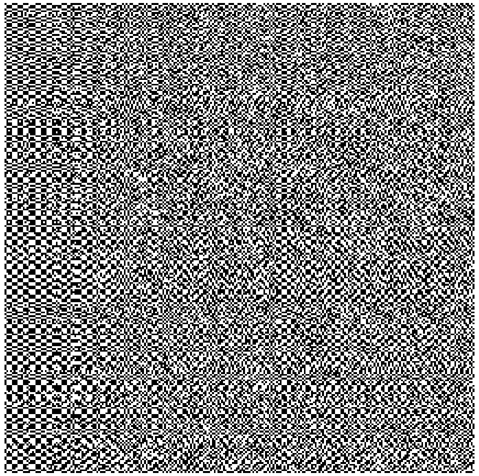
Can we simply take it?

Image deblurring problem: Original image x, noisy blurred image b and the "naive" solution $x^{\text{naive}} \equiv A^{-1}b$:

x = true image

b = blurred, noisy image

x = inverse solution



Why it does not work? Because of the properties of our problem.

Consider that b^{noise} is noise and b^{exact} is the exact part in our image b. Then our linear model is

$$Ax = b, \quad b = b^{\text{exact}} + b^{\text{noise}}.$$

Usual properties:

- the problem is ill-posed (i.e. sensitive to small changes in b);
- often A is nonsingular, but its singular values σ_j decay quickly;
- b^{exact} is smooth, and satisfies the discrete Picard condition (DPC);
- b^{noise} is often random - white (does not have dominant frequencies), and does not satisfy DPC;
- $\|b^{\text{exact}}\| \gg \|b^{\text{noise}}\|$, but $\|A^{-1}b^{\text{exact}}\| \ll \|A^{-1}b^{\text{noise}}\|$.

SVD components of the naive solution:

From the SVD $A = U \Sigma V^T$, we have

$$A^{-1} = V \Sigma^{-1} U^T = \sum_{j=1}^{n} v_j \frac{1}{\sigma_j} u_j^T = \sum_{j=1}^{n} \frac{1}{\sigma_j} v_j \, u_j^T \, .$$

Thus

$$\begin{aligned} x^{\text{naive}} \; &\equiv \; A^{-1} b \; = \; \sum_{j=1}^{n} \frac{1}{\sigma_j} v_j \, u_j^T b \; = \; \sum_{j=1}^{n} \frac{u_j^T b}{\sigma_j} v_j \\ &= \; \underbrace{\sum_{j=1}^{n} \frac{u_j^T b^{\text{exact}}}{\sigma_j} v_j}_{x^{\text{exact}} = A^{-1} b^{\text{exact}}} + \underbrace{\sum_{j=1}^{n} \frac{u_j^T b^{\text{noise}}}{\sigma_j} v_j}_{A^{-1} b^{\text{noise}}} \, . \end{aligned}$$

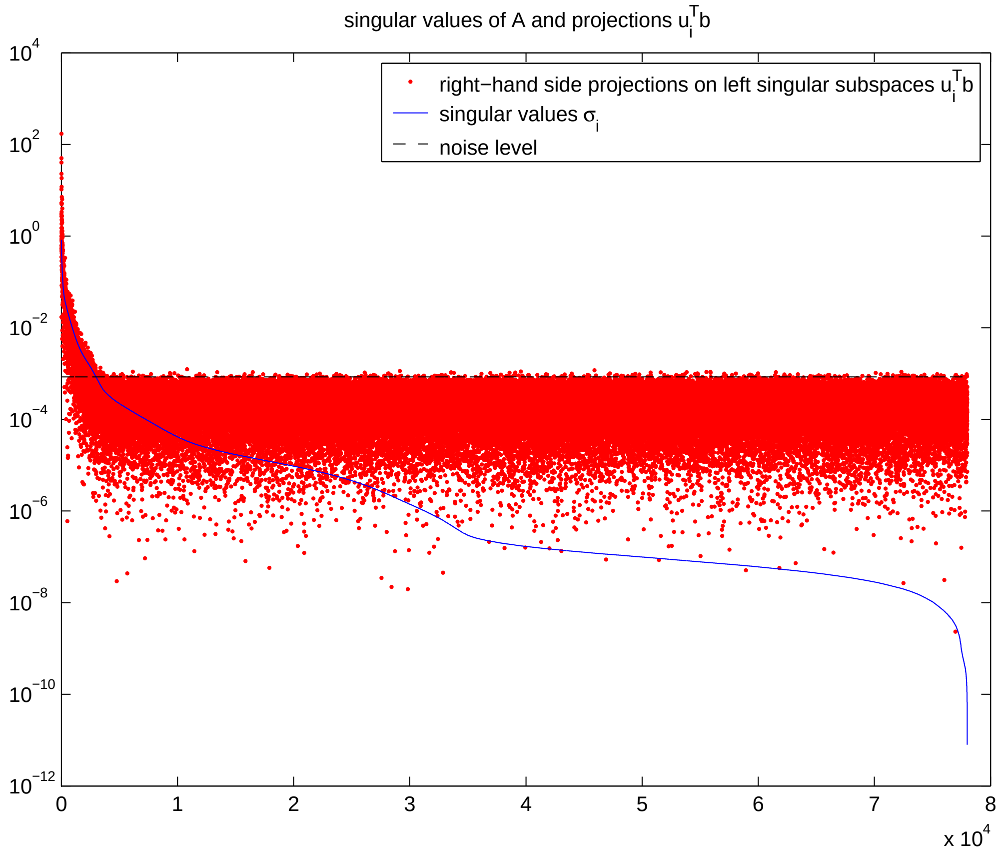
$$x^{\text{naive}} = \underbrace{\sum_{j=1}^{n} \frac{u_j^T b^{\text{exact}}}{\sigma_j} v_j}_{x^{\text{exact}} = A^{-1} b^{\text{exact}}} + \underbrace{\sum_{j=1}^{n} \frac{u_j^T b^{\text{noise}}}{\sigma_j} v_j}_{A^{-1} b^{\text{noise}}}$$

We want the left sum x^{exact}. What is the size of the right sum (inverted noise) in comparison to the left one?

Exact data satisfy DPC:
On average, $|u_j^T b^{\text{exact}}|$ decay faster than the singular values σ_j of A.

White noise does not:
The values $|u_j^T b^{\text{noise}}|$ do not exhibit any trend.

Thus $u_j^T b = u_j^T b^{\text{exact}} + u_j^T b^{\text{noise}}$ are for small indexes j dominated by the exact part, but for large j by the noisy part.



Because of the division by the singular value, the components of the naive solution

$$x^{\text{naive}} \equiv A^{-1}b = \underbrace{\sum_{j=1}^{k} \frac{u_j^T b^{\text{exact}}}{\sigma_j} v_j}_{x^{\text{exact}}} + \underbrace{\sum_{j=1}^{k} \frac{u_j^T b^{\text{noise}}}{\sigma_j} v_j}_{\text{inverted noise}}$$

$$+ \underbrace{\sum_{j=k+1}^{N} \frac{u_j^T b^{\text{exact}}}{\sigma_j} v_j}_{x^{\text{exact}}} + \underbrace{\sum_{j=k+1}^{N} \frac{u_j^T b^{\text{noise}}}{\sigma_j} v_j}_{\text{inverted noise}}$$

corresponding to small σ_j's are dominated by inverted noise.

Remember, there are usually many small singular values.

Basic regularization method - Truncated SVD:

Using the dyadic form

$$A = U\,\Sigma\,V^T = \sum_{i=1}^{n} u_i \sigma_i v_i^T,$$

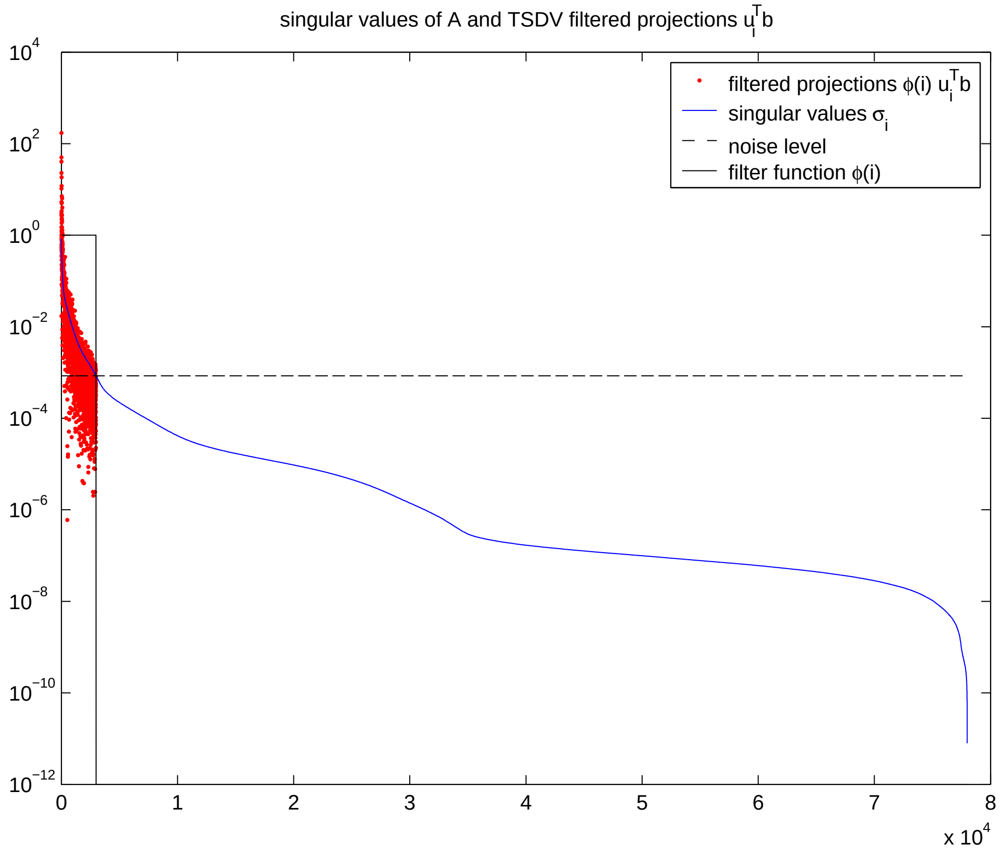
we can approximate A with a rank k matrix

$$A \approx S_k \equiv \sum_{i=1}^{k} A_i = \sum_{i=1}^{k} u_i\,\sigma_i\,v_i^T.$$

Replacing A by S_k gives an TSVD approximate solution

$$x^{(k)} = \sum_{j=1}^{k} \frac{u_j^T b}{\sigma_j} v_j\,.$$

TSVD regularization: removing of troublesome components



Here the smallest σ_j's are not present. However, we removed also some components of x^{exact}.

An optimal k has to balance between removing noise and not loosing too many components of the exact solution. It depends on the matrix properties and on the amount of noise in the considered image.

MATLAB DEMO: Compute TSVD regularized solutions for different values of k. Compare quality of the obtained image.

Other regularization methods:

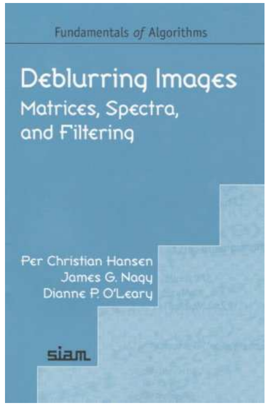
- direct regularization;
- stationary regularization;
- projection (including iterative) regularization;
- hybrid methods combining the previous ones.

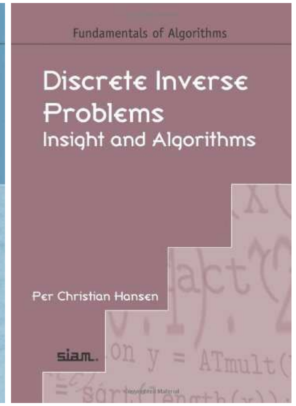
Other applications:

- computer tomography (CT);
- magnetic resonance;
- seismology;
- crystallography;
- material sciences;
- ...

References:

Textbooks:





- Hansen, Nagy, O'Leary: *Deblurring Images, Spectra, Matrices, and Filtering*, SIAM, 2006.
- Hansen: *Discrete Inverse Problems, Insight and Algorithms*, SIAM, 2010.

Software (MatLab toolboxes): on the homepage of P. C. Hansen

- HNO package,
- Regularization tools,
- AIRtools,
- ...