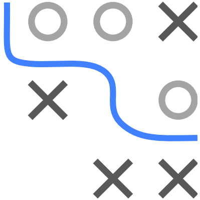
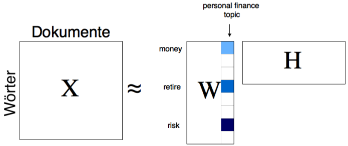
# Algorithms and Data Structures



## Matrix Approximation
## Low-Rank Approximation




### Learning goals

- Low-Rank Approximation

# LOW-RANK APPROXIMATION

Let $\mathbf{X}$ be a $m \times n$ data matrix, where the columns of the matrix represent different "objects" (images, text documents, ...). In many practical applications $\mathbf{X}$ is high-dimensional.

| Data | Columns | Rows | $m$ | $n$ |
|---|---|---|---|---|
| Image data | Images | Pixel intensities | $> 10^8$ | $10^5 - 10^6$ |
| Text data | Text documents | Word frequencies | $10^5 - 10^7$ | $> 10^{10}$ |
| Product reviews | Products | User reviews | $10^1 - 10^4$ | $> 10^7$ |
| Audio data$^{(*)}$ | Points in time | Strength of a frequency | $10^5 - 10^6$ | $> 10^8$ |

$^{(*)}$ Example: `https://musiclab.chromeexperiments.com/Spectrogram`

# LOW-RANK APPROXIMATION / 2

In a low-rank approximation, **X** is factorized into two matrices $\mathbf{W} \in \mathbb{R}^{m \times k}$ and $\mathbf{H} \in \mathbb{R}^{k \times n}$ such that

$$\mathbf{X} \approx \underbrace{\mathbf{W}}_{\substack{\text{"dictionary",}\\ \text{"patterns",}\\ \text{"topics"}}} \cdot \underbrace{\mathbf{H}}_{\text{"regressors"}}$$

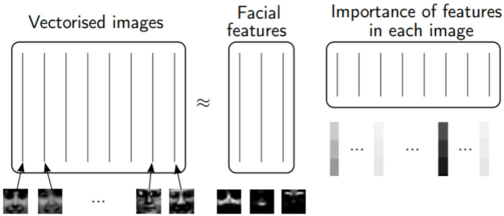
Compared to $n$ and $m$, $k$ is usually small.

# LOW-RANK APPROXIMATION / 3

**Introductory Example 1:** Image Processing(*)

Given are *n* images in vectorized form.

$$\boldsymbol{X} \quad = \quad \boldsymbol{W} \quad \cdot \quad \boldsymbol{H}$$




(*) Example from `http://perso.telecom-paristech.fr/~essid/teach/NMF_tutorial_ICME-2014.pdf`

# LOW-RANK APPROXIMATION / 4

**(Possible) Advantages:**

- The dimension reduction reveals **latent variables** (here: "Facial Features") and the data can be "explained".
- The storage space can be reduced significantly (for appropriate choice of $k$). Instead of a $m \times n$ matrix, a $m \times k$ and a $k \times n$ matrix with $k \ll m, n$ must be stored.

  Calculation example: $n = 1000$ images with $m = 10000$ pixels each. Using a matrix approximation of rank 10 the storage space can be reduced from $m \times n = 1 \times 10^6$ to $m \times k + k \times n = 10000 \cdot 10 + 10 \cdot 1000 = 110000$ (about $10\%$ of the original size).

# LOW-RANK APPROXIMATION / 5

**Introductory Example 2:** Text mining

Given is a $m \times n$ document-term matrix $\mathbf{X}$, where

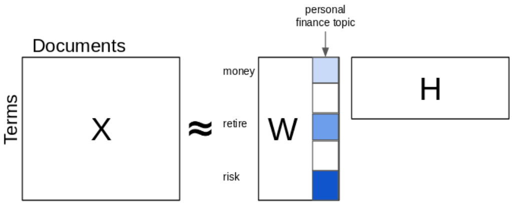
$$x_{ij} = \text{Frequency of term } i \text{ in document } j$$

Using a low-rank approximation, we approximate $\mathbf{X}$ with

$$\mathbf{X} \approx \mathbf{WH}$$

Suppose we want to display various newspaper articles in a document-term matrix.

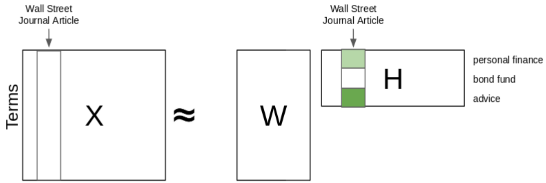
# LOW-RANK APPROXIMATION / 6

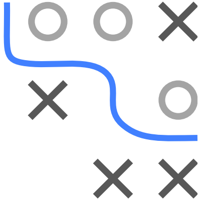



The $k$ columns in $\mathbf{W}$ represent different topics, and the entries of $\mathbf{W}$ can be interpreted as

$$w_{ij} = \text{connection of word } i \text{ and subject } j$$

# LOW-RANK APPROXIMATION / 7

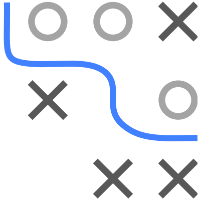


The entries of **H** can be interpreted as

$$h_{ij} = \text{Measure for how much article } j \text{ discusses topic } i$$

# LOW-RANK APPROXIMATION / 8



For fixed $k$ this can be formulated as a general optimization problem

$$\min_{\mathbf{W}\in\mathbb{R}^{m\times k},\mathbf{H}\in\mathbb{R}^{k\times n}} \|\mathbf{X}-\mathbf{WH}\|_F^2$$

The Eckart-Young-Mirsky theorem states that the solution of the optimization problem is given by the **truncated singular value decomposition**

$$\mathbf{X} \approx \boldsymbol{W}\boldsymbol{H} = \mathbf{U}_k\Sigma_k\mathbf{V}_k^\top$$

where matrix $\Sigma_k$ contains the $k$ largest **singular values** and the matrices $\mathbf{U}_k$, $\mathbf{V}_k$ contain the corresponding **singular vectors** of **X**.

The matrices $\boldsymbol{W}$ and $\boldsymbol{H}$ can be set as $\boldsymbol{W} := \mathbf{U}_k\Sigma_k$ and $\boldsymbol{H} := \mathbf{V}_k^\top$ or as $\boldsymbol{W} := \mathbf{U}_k(\Sigma_k)^{1/2}$ and $\boldsymbol{H} := (\Sigma_k)^{1/2}\mathbf{V}_k^\top$.