# Bi7740: Scientific computing

## Matrix computation - Exercises

Vlad Popovici
popovici@iba.muni.cz

Institute of Biostatistics and Analyses
Masaryk University, Brno

# Before starting

Let

- $\mathbf{x} \in \mathbb{R}^m$, $\mathbf{x} = [x_1, \ldots, x_m]$
- $\mu \in \mathbb{R}^m$ be the mean (or sample average, depending on the context) vector
- $\mathbf{X} = \begin{bmatrix} \mathbf{x}_1 \\ \vdots \\ \mathbf{x}_n \end{bmatrix}$ and $\mathbf{X}_\mu = \begin{bmatrix} \mathbf{x}_1 - \mu \\ \vdots \\ \mathbf{x}_n - \mu \end{bmatrix}$
- $\Sigma = E[(\mathbf{x} - \mu)^t (\mathbf{x} - \mu)]$ the covariance matrix
- $\hat{\Sigma} = \frac{1}{n-1} \mathbf{X}_\mu^t \mathbf{X}_\mu$ the sample-based unbiased estimator of the covariance
- load data: `load 'artificial_data.mat'` and `load 'genexpr_data.mat'`

# Outline

1. Mahalanobis distance
2. Principal component analysis
3. Data sphering

# Mahalanobis distance

Let $\mathcal{N}(\mu, \Sigma)$ be a *m*-dimensional Gaussian (normal) distribution with mean $\mu$ and covariance $\Sigma$. Then,

**Mahalanobis distance**

$$d^2(\mathbf{x}, \mathcal{N}(\mu, \Sigma)) = (\mathbf{x} - \mu)\Sigma^{-1}(\mathbf{x} - \mu)^t$$

# Exercise

Write the MATLAB functions for computing the Mahalanobis distance from each of the observations (rows) in **X** under a Gaussian distribution:

- `mhdist0(X)` for $\mathcal{N}(0, \mathbf{I})$ (Euclidean distance to **0**!)
- `mhdist1(X, mu, sigma)` for $\mathcal{N}(\mu, \Sigma)$ with $\mu$ and $\Sigma$ provided by the user
- `mhdist(X, Z)` for $\mathcal{N}(\hat{\mu}, \hat{\Sigma})$ with $\hat{\mu}$ and $\hat{\Sigma}$ estimated from $Z$

# Outline

# PCA

- $\mathbf{Z} = \mathbf{X} * \mathbf{W}$
- an orthogonal transformation such that the new axes align with the directions of lagest variation
- the resulting variables are linearly uncorrelated
- one interpretation: finds the axes (linear combinations of original variables) that minimize the squared-error of approximating the original data (linear regression)
- other perspective: spectral analysis of the covariance matrix
- **W**: has the columns the eigenvectors of the covariance matrix of vectors in **X**

# Exercise

Write MATLAB functions

- `prcomp_naive(X, npc)`: computes the principal vectors and corresponding coefficients by eigendecomposition of the covariance matrix
- what happens when trying to find all the eigenvectors for $m > n$? What if $m >> n$?
- remember that SVD decomposition of $\mathbf{X}$ is mathematically equivalent to eigendecomposition of $\mathbf{X}^t\mathbf{X}$. Implement `prcomp(X)` to find all principal vectors and corresponding coefficients, by using the SVD decomposition.

# Outline

1. Mahalanobis distance
2. Principal component analysis
3. Data sphering

# Data sphering

- PCA can be used to uncorrelated variables
- by proper scaling of the result, the resulting variables have identity covariance matrix:

$$\mathbf{Z} = \mathbf{X}\left(\mathbf{V}\Lambda^{-\frac{1}{2}}\right)$$

  where Var has the eigenvectors of covariance matrix as columns and $\Lambda$ is the diagonal matrix of corresponding eigenvalues.
- it is also called data whitening because the spectrum of eigenvalues of the transformed (distribution) uniform

# Exercise

Implement in MATLAB the data sphering (DO NOT USE `inv()`):

- `[Z, W, IW] = sphere(X)` transforms the data $X \longrightarrow Z$, and returns the transformation matrix $W$ and its inverse $IW$
- try it!
- `[Z,W] = sphere2(X)` for matrices with $m >> n$: use the following math:

$$\mathbf{X}\mathbf{X}^t\mathbf{u} = \lambda\mathbf{u} \Leftrightarrow (\mathbf{X}^t\mathbf{X})(\mathbf{X}^t\mathbf{u}) = \lambda\mathbf{X}^t\mathbf{u}$$

  so the eigenvectors $\mathbf{v}_i$ of the covariance matrix of interest (of the form $\mathbf{X}^t\mathbf{X}$) can be obtained from the eigenvectors $\mathbf{u}_i$ of $\mathbf{X}\mathbf{X}^t$ by $\mathbf{v}_i = \mathbf{X}'\mathbf{u}_i$ followed by proper scaling (for normalization)