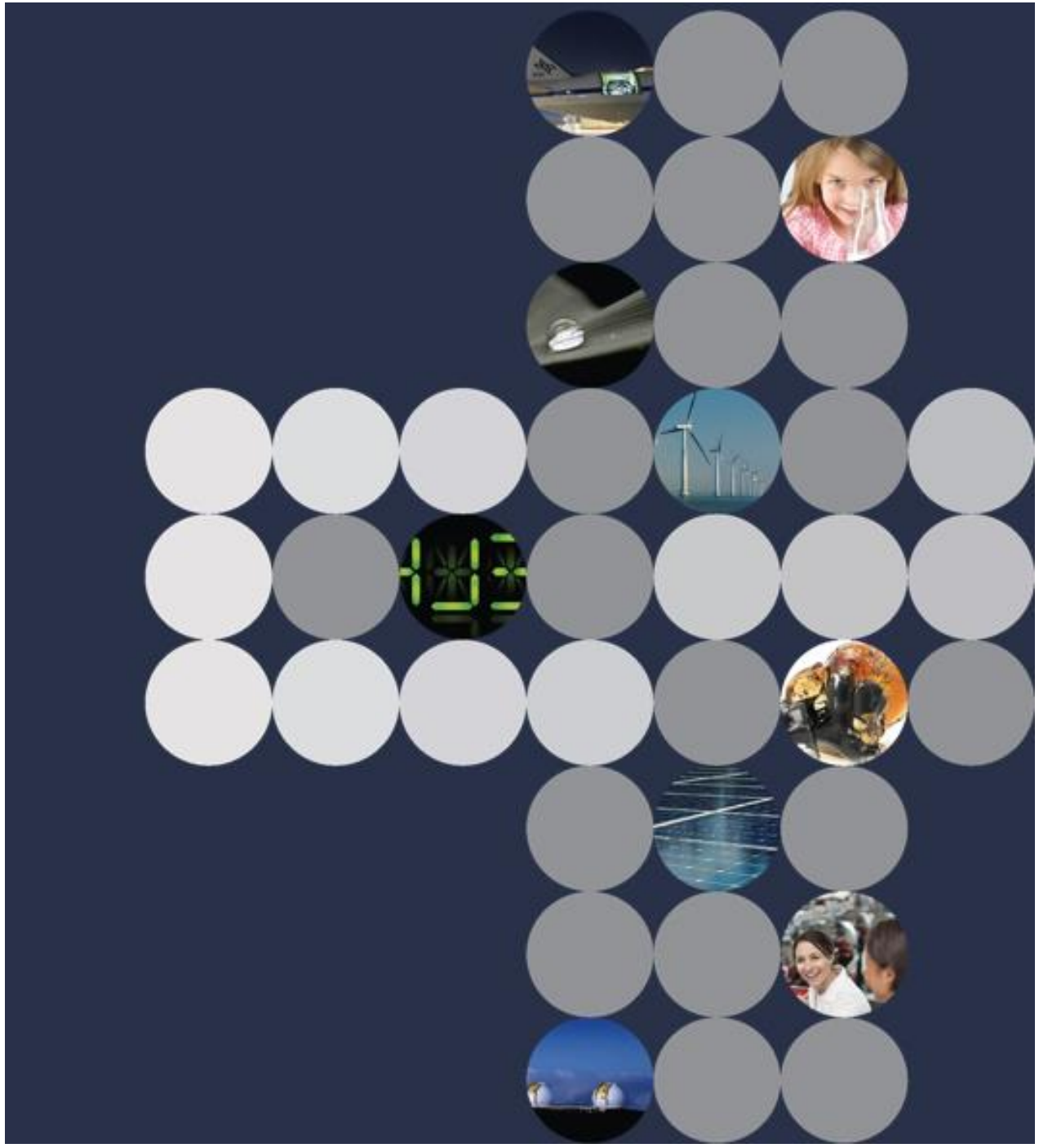
EE512 – Applied Biomedical Signal Processing

Principal Component Analysis (PCA)

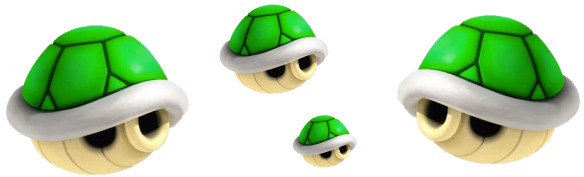
Karen ADAM
CSEM Signal Processing and AI Group



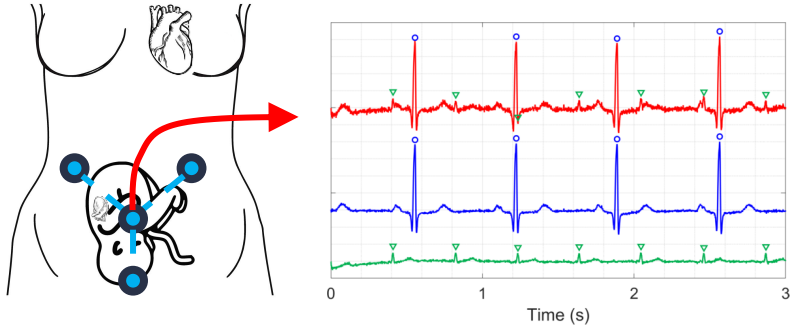
EPFL :: csem

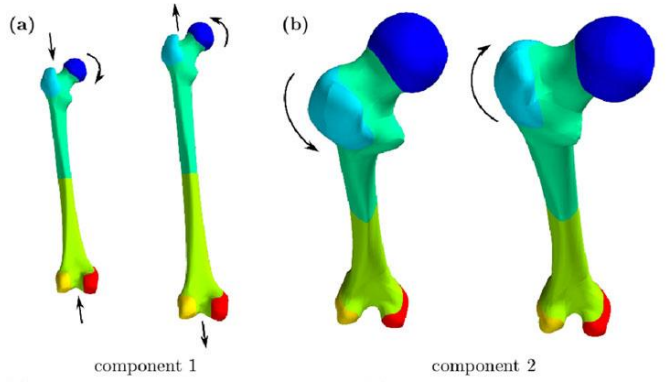
Introduction - Principal Component Analysis (PCA)

- Principal component analysis (PCA)
  - Also called Hotelling or Karhunen-Loève transform



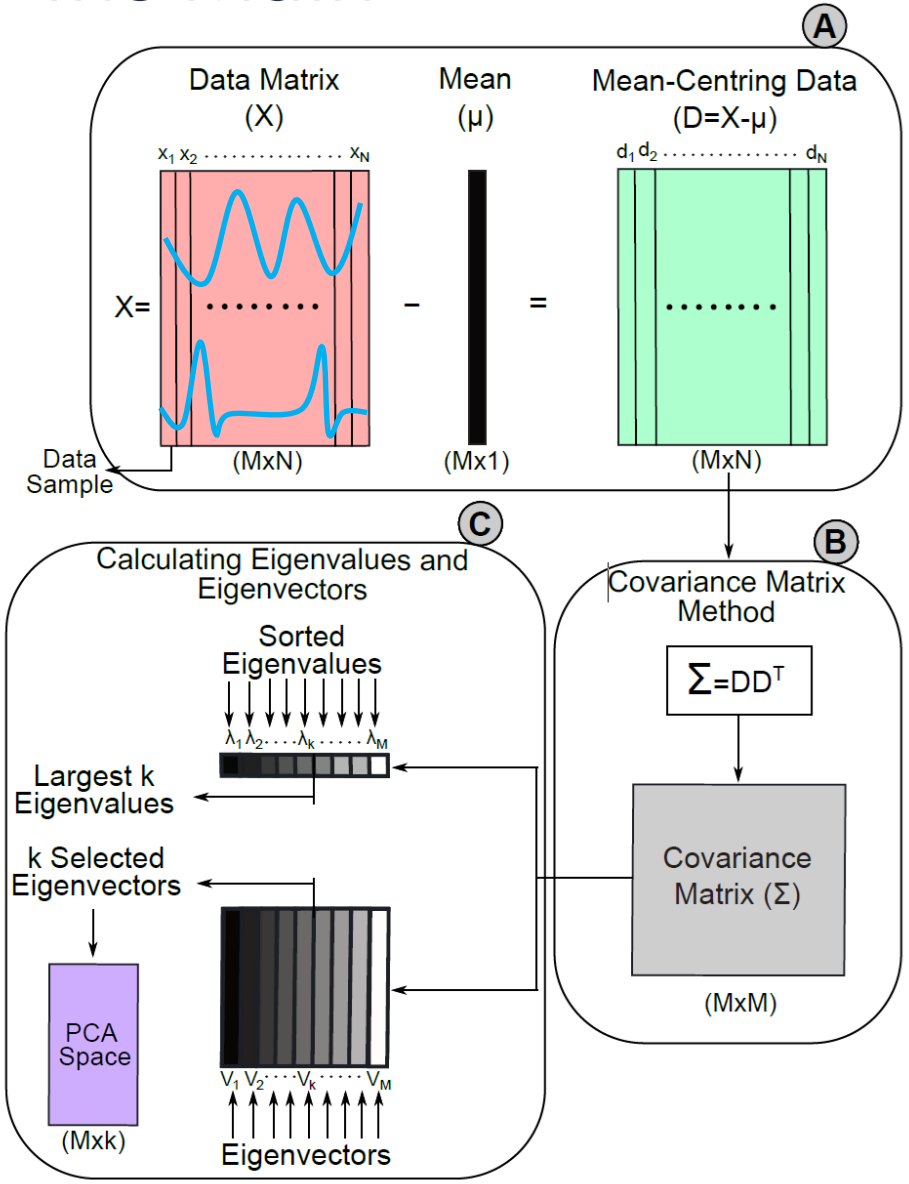
- Motivation: use-cases
  1. Dimensionality reduction
  2. Blind source separation
  3. Statistical shape analysis/modelling





EPFL :: csem

The Math



- Data matrix X of size $M \times N$
  - M: dimensions (e.g., # channels)
  - N: trials/experiments (e.g., # samples)
- 4-step approach:
  A. Remove mean: $\boldsymbol{D} = \boldsymbol{X} - \mu$
  B. Covariance matrix: $\boldsymbol{\Sigma} = \boldsymbol{D}\boldsymbol{D}^T$
  C. Eigenvectors v_i and Eigenvalues λ_i:

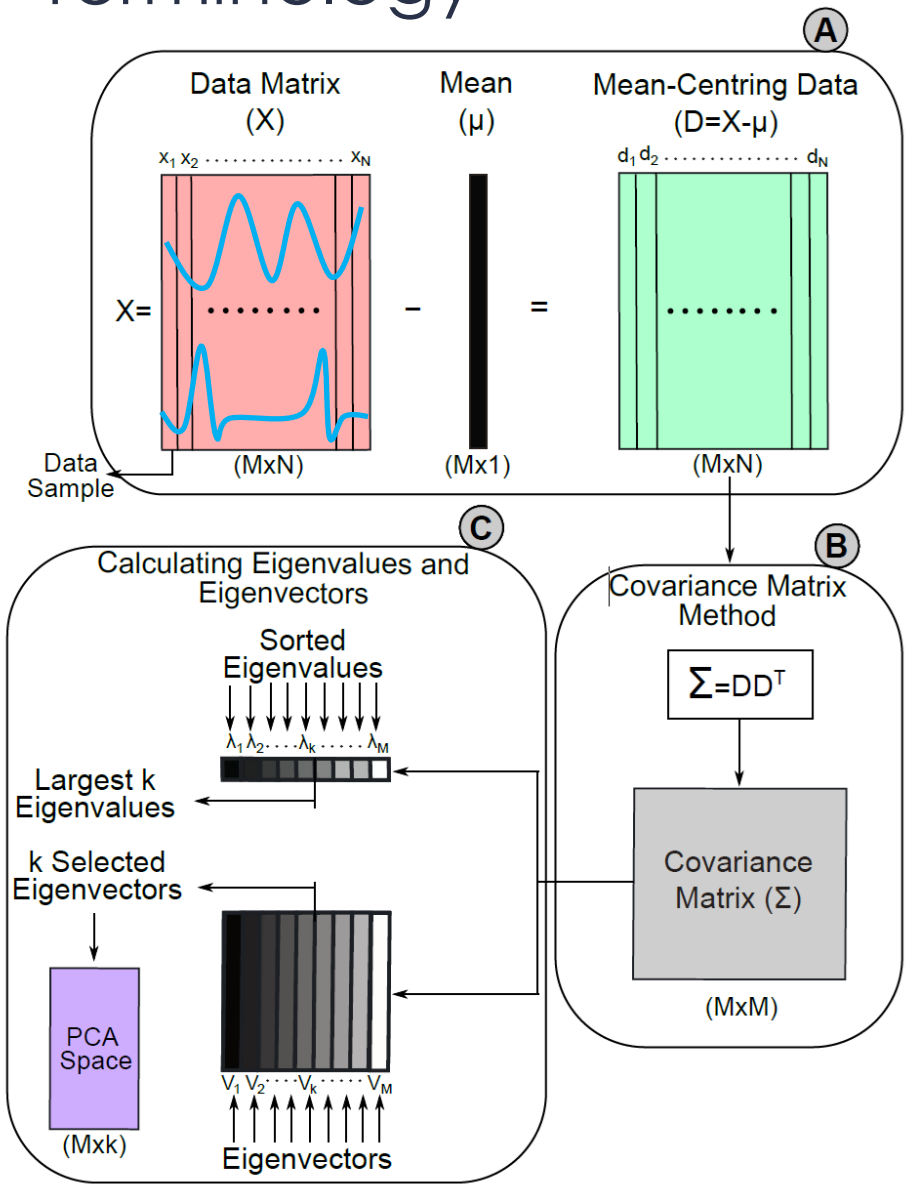
$$\boldsymbol{W} = \{v_1, v_2, \dots, v_M\} \\ \{\lambda_1, \lambda_2, \dots, \lambda_M\} \Leftarrow \text{eig}\,(\boldsymbol{\Sigma})$$

  D. Principal components:

$$\mathbf{Y} = \mathbf{W}^T \boldsymbol{D} \in \mathbb{R}^{M \times N}$$

A. Tharwat, "Principal component analysis - a tutorial," *IJAPR*, vol. 3, no. 3, p. 197, 2016, doi: 10.1504/IJAPR.2016.079733.

Terminology



- Data matrix X of size $M \times N$
  - M: dimensions (e.g., # channels)
  - N: trials/experiments (e.g., # samples)
- 4-step approach:
  A. Remove mean: $\boldsymbol{D} = \boldsymbol{X} - \mu$
  B. Covariance matrix: $\boldsymbol{\Sigma} = \boldsymbol{D}\boldsymbol{D}^T$
  C. Eigenvectors v_i and Eigenvalues λ_i:

$$\boldsymbol{W} = \{v_1, v_2, \dots, v_M\}$$
$$\{\lambda_1, \lambda_2, \dots, \lambda_M\}$$
$$\Leftarrow \text{eig}(\boldsymbol{\Sigma})$$

- *Principal axes*
- *Principal directions*
- ~~*Principal components (!)*~~

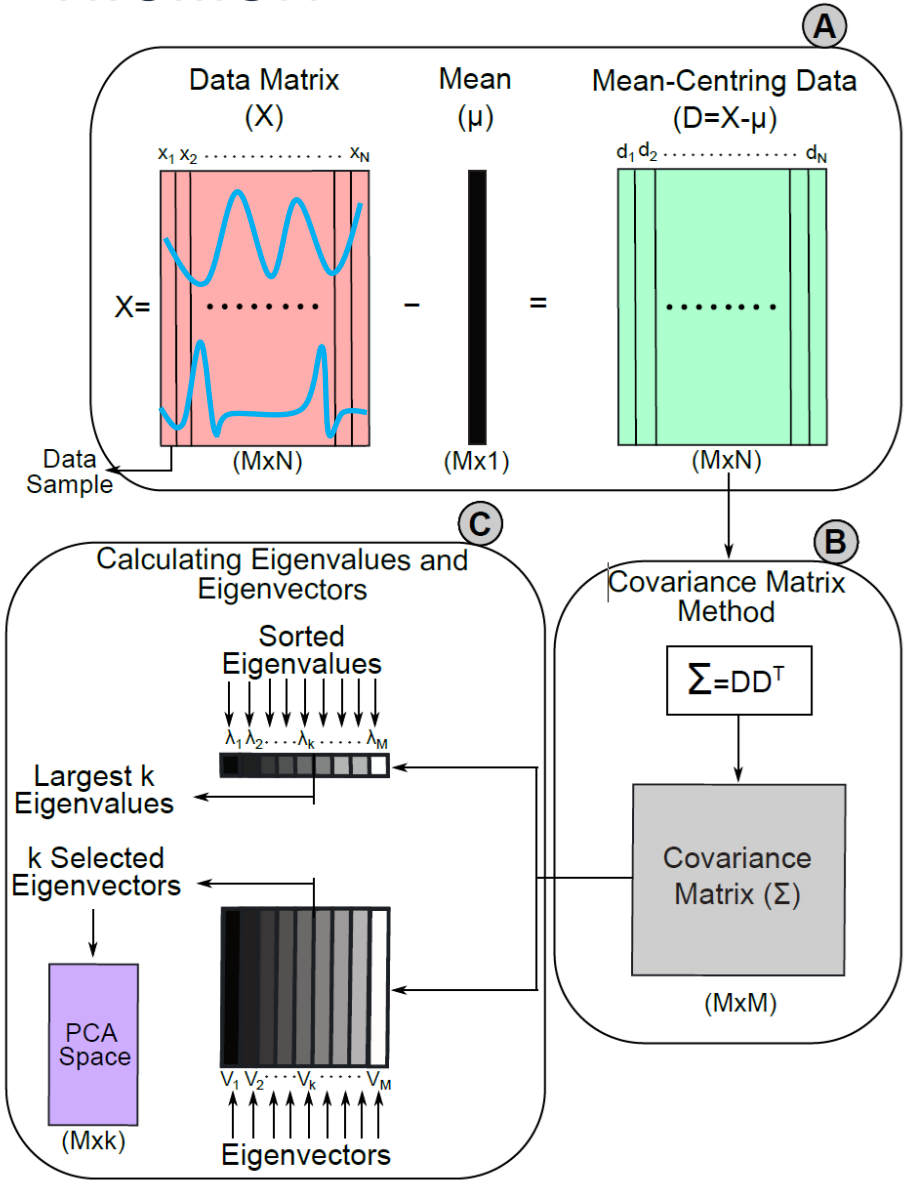
  D. Principal components:

$$\mathbf{Y} = \mathbf{W}^T \boldsymbol{D} \in \mathbb{R}^{M \times N}$$

- *Principal component scores*

A. Tharwat, "Principal component analysis - a tutorial," *IJAPR*, vol. 3, no. 3, p. 197, 2016, doi: 10.1504/IJAPR.2016.079733.

Intuition



What you'd like to reduce

- Data matrix X of size $M \times N$
  - M: dimensions (e.g., # channels)
  - N: trials/experiments (e.g., # samples)
- 4-step approach:
  A. Remove mean: $\boldsymbol{D} = \boldsymbol{X} - \mu$
  B. Covariance matrix: $\boldsymbol{\Sigma} = \boldsymbol{D}\boldsymbol{D}^T$
  C. Eigenvectors v_i and Eigenvalues λ_i:

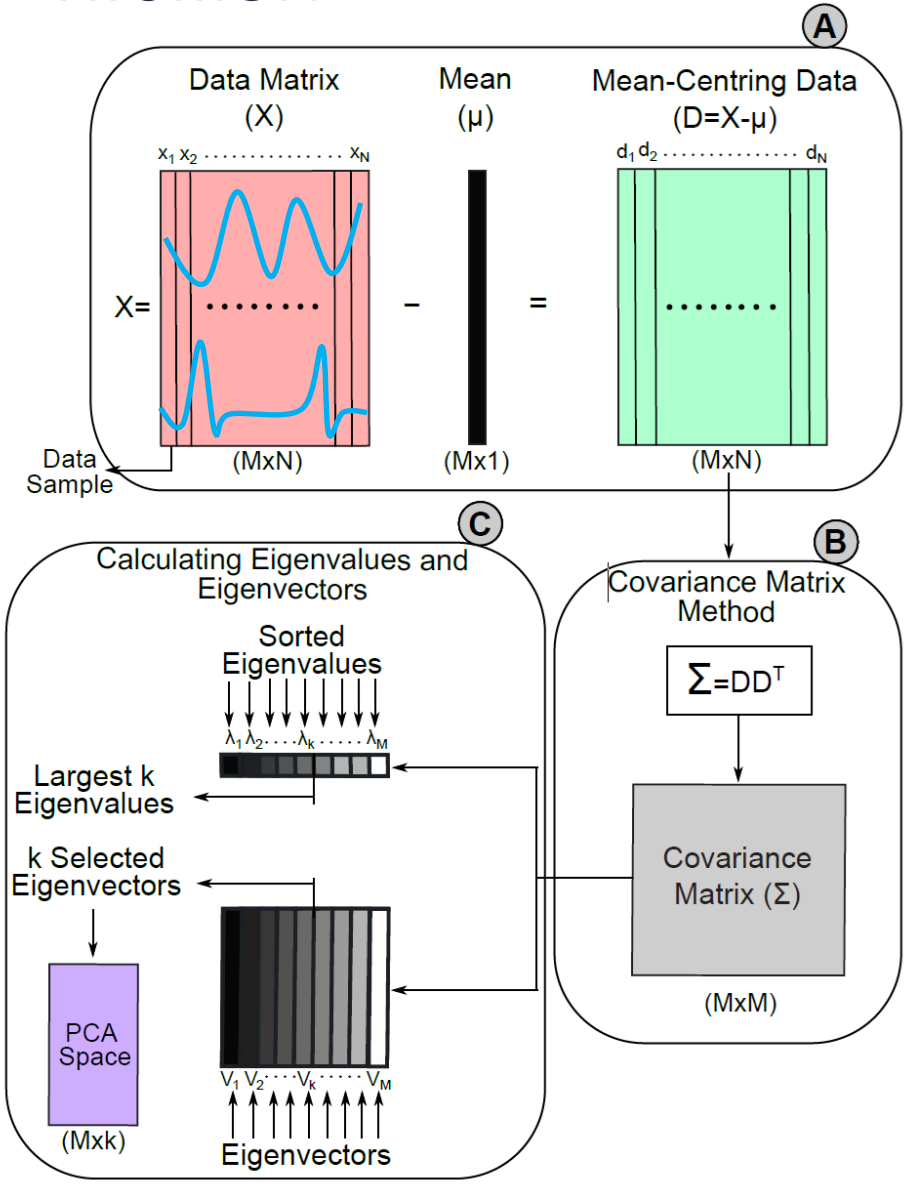
$$\boldsymbol{W} = \{v_1, v_2, \dots, v_M\}$$
$$\{\lambda_1, \lambda_2, \dots, \lambda_M\} \Leftarrow \text{eig}\,(\boldsymbol{\Sigma})$$

  D. Principal components:

$$\mathbf{Y} = \mathbf{W}^T \boldsymbol{D} \in \mathbb{R}^{M \times N}$$

A. Tharwat, "Principal component analysis - a tutorial," *IJAPR*, vol. 3, no. 3, p. 197, 2016, doi: 10.1504/IJAPR.2016.079733.

Intuition



- Data matrix X of size $M \times N$

What you'd like to reduce

  - M: dimensions (e.g., # channels)
  - N: trials/experiments (e.g., # samples)

- 4-step approach:

A. Remove mean: $\boldsymbol{D} = \boldsymbol{X} - \mu$

B. Covariance matrix: $\boldsymbol{\Sigma} = \boldsymbol{D}\boldsymbol{D}^T$

C. Eigenvectors v_i and Eigenvalues λ_i:

MxM rotation matrix in decreasing λ order

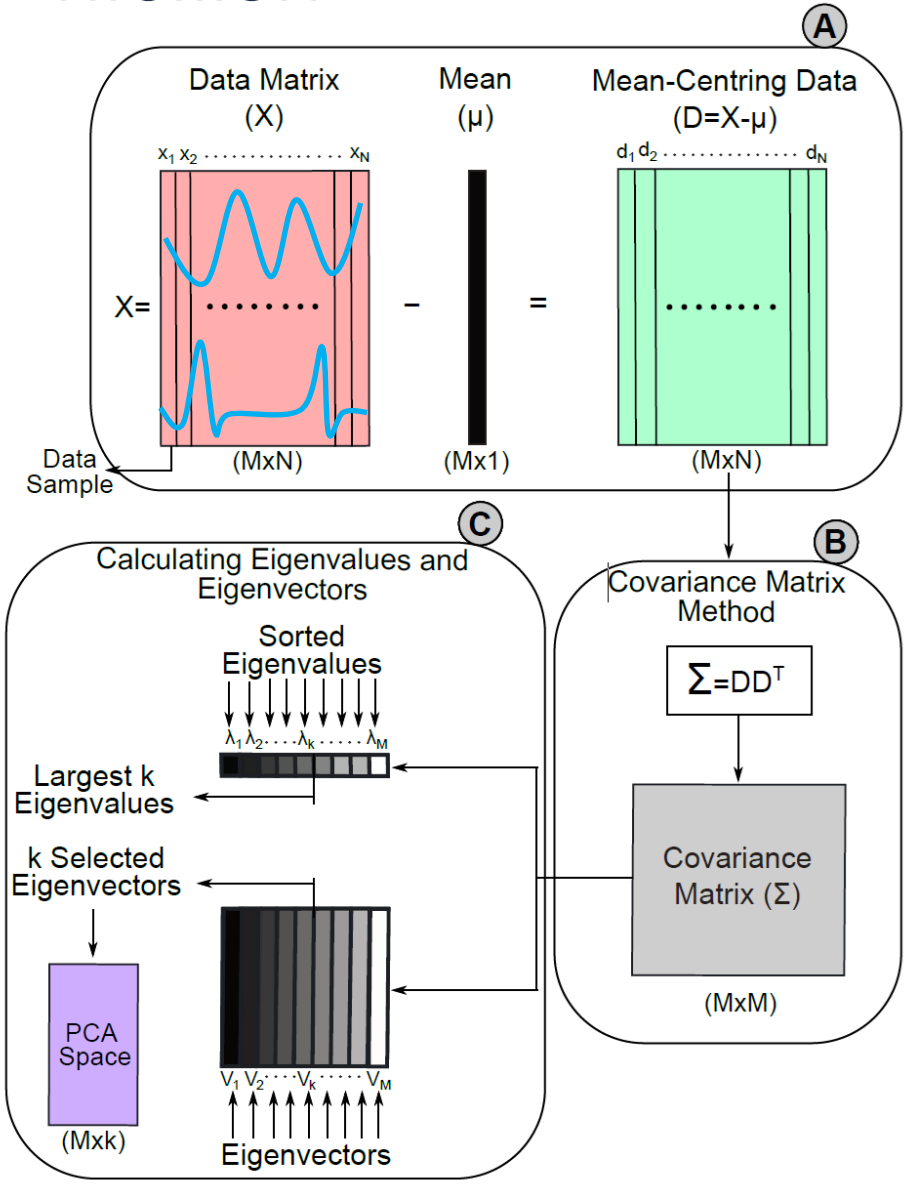
$$\boldsymbol{W} = \{v_1, v_2, \dots, v_M\} \Leftarrow \text{eig}\,(\boldsymbol{\Sigma})$$
$$\{\lambda_1, \lambda_2, \dots, \lambda_M\}$$

D. Principal components:

$$\mathbf{Y} = \mathbf{W}^T \boldsymbol{D} \in \mathbb{R}^{M \times N}$$

A. Tharwat, "Principal component analysis - a tutorial," *IJAPR*, vol. 3, no. 3, p. 197, 2016, doi: 10.1504/IJAPR.2016.079733.

Intuition



- Data matrix X of size $M \times N$

What you'd like to reduce

  - M: dimensions (e.g., # channels)
  - N: trials/experiments (e.g., # samples)

- 4-step approach:

A. Remove mean: $\boldsymbol{D} = \boldsymbol{X} - \mu$

B. Covariance matrix: $\boldsymbol{\Sigma} = \boldsymbol{D}\boldsymbol{D}^T$

C. Eigenvectors v_i and Eigenvalues λ_i:

MxM rotation matrix in decreasing λ order

$$\boldsymbol{W} = \{v_1, v_2, \dots, v_M\} \Leftarrow \text{eig}(\boldsymbol{\Sigma})$$
$$\{\lambda_1, \lambda_2, \dots, \lambda_M\}$$

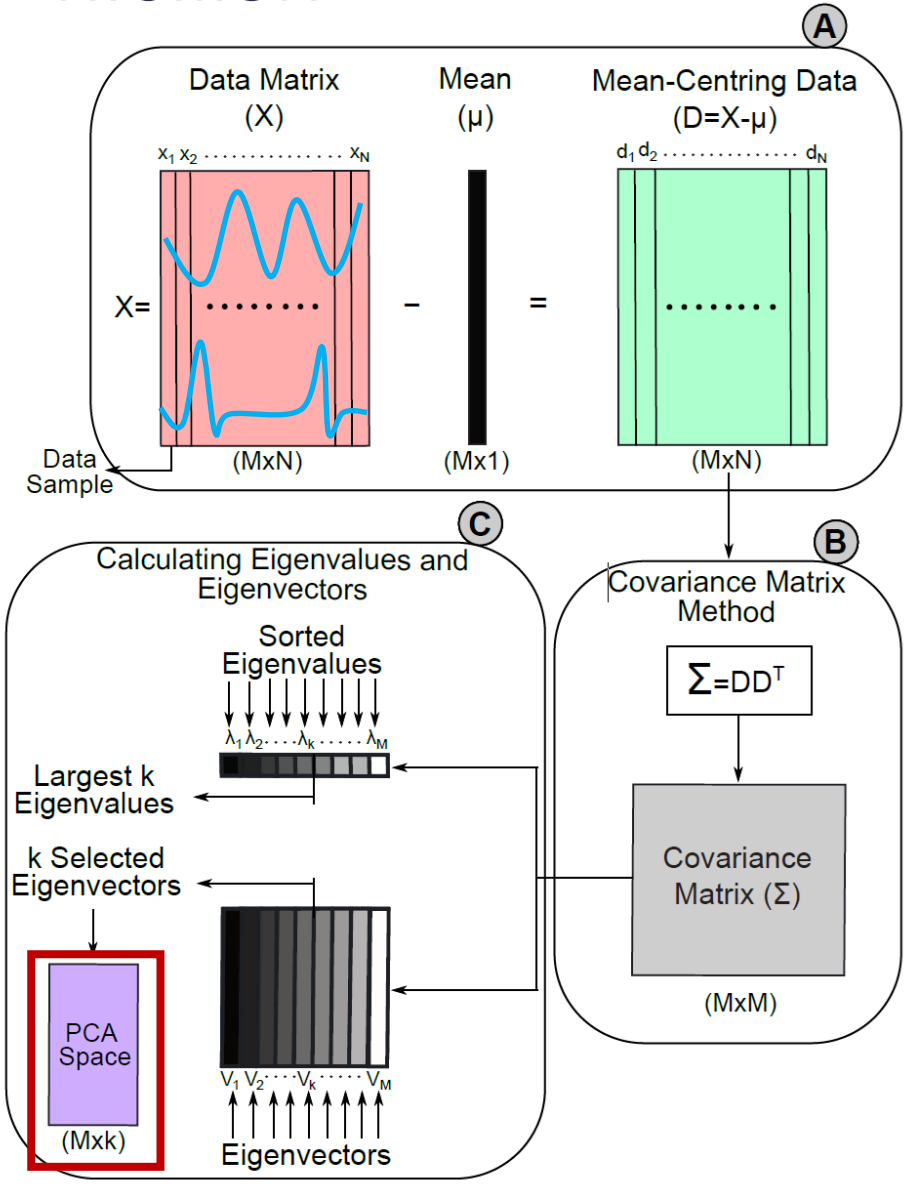
D. Principal components:

The rotated data matrix

$$\mathbf{Y} = \mathbf{W}^T \boldsymbol{D} \in \mathbb{R}^{M \times N}$$

A. Tharwat, "Principal component analysis - a tutorial," *IJAPR*, vol. 3, no. 3, p. 197, 2016, doi: 10.1504/IJAPR.2016.079733.

Intuition



- Data matrix X of size $M \times N$

What you'd like to reduce

  - M: dimensions (e.g., # channels)
  - N: trials/experiments (e.g., # samples)

- 4-step approach:

A. Remove mean: $\boldsymbol{D} = \boldsymbol{X} - \mu$

B. Covariance matrix: $\boldsymbol{\Sigma} = \boldsymbol{D}\boldsymbol{D}^T$

C. Eigenvectors v_i and Eigenvalues λ_i:

MxM rotation matrix in decreasing λ order

$$\boldsymbol{W} = \{v_1, v_2, \dots, v_M\} \Leftarrow \text{eig}(\boldsymbol{\Sigma})$$
$$\{\lambda_1, \lambda_2, \dots, \lambda_M\}$$

D. Principal components:

The rotated data matrix

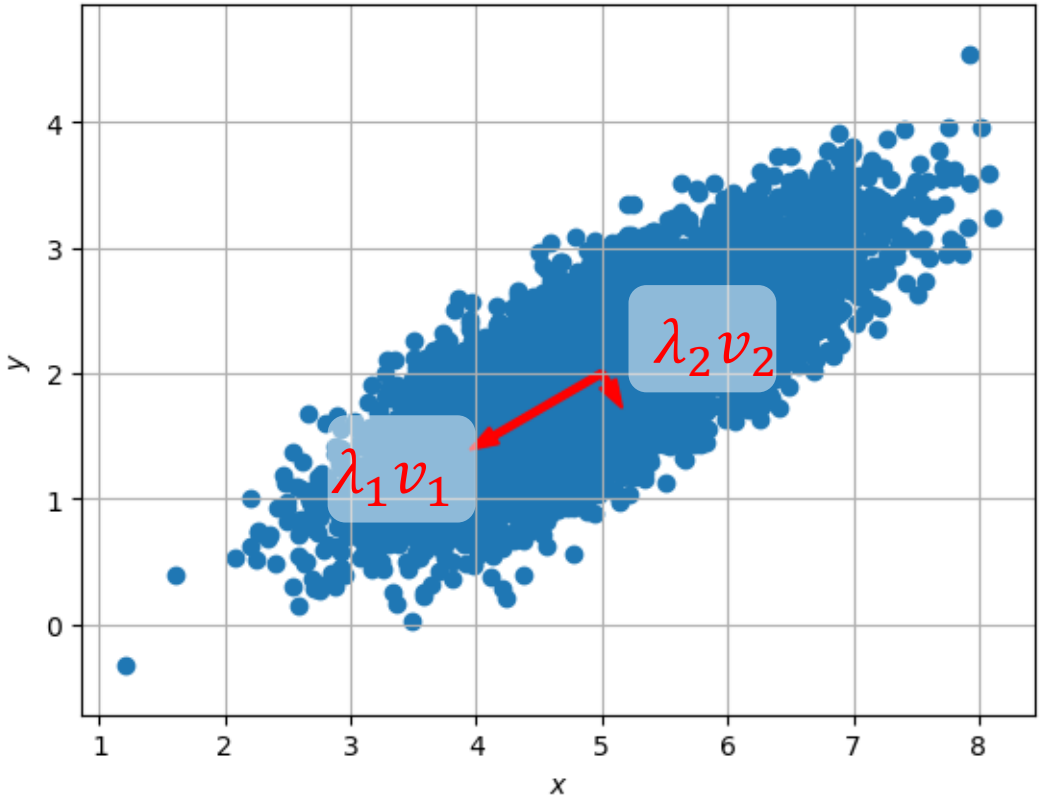
$$\mathbf{Y} = \mathbf{W}^T \boldsymbol{D} \in \mathbb{R}^{M \times N}$$

A. Tharwat, "Principal component analysis - a tutorial," *IJAPR*, vol. 3, no. 3, p. 197, 2016, doi: 10.1504/IJAPR.2016.079733.

Simple Examples

$$\mathbf{X} = \underbrace{\begin{bmatrix} x_1 & x_2 & \cdots & x_N \\ y_1 & y_2 & \cdots & y_N \end{bmatrix}}_{\text{N = 10'000 \# samples}} \Big\} \text{M = 2 \# dimensions}$$

- 2D Gaussian distribution



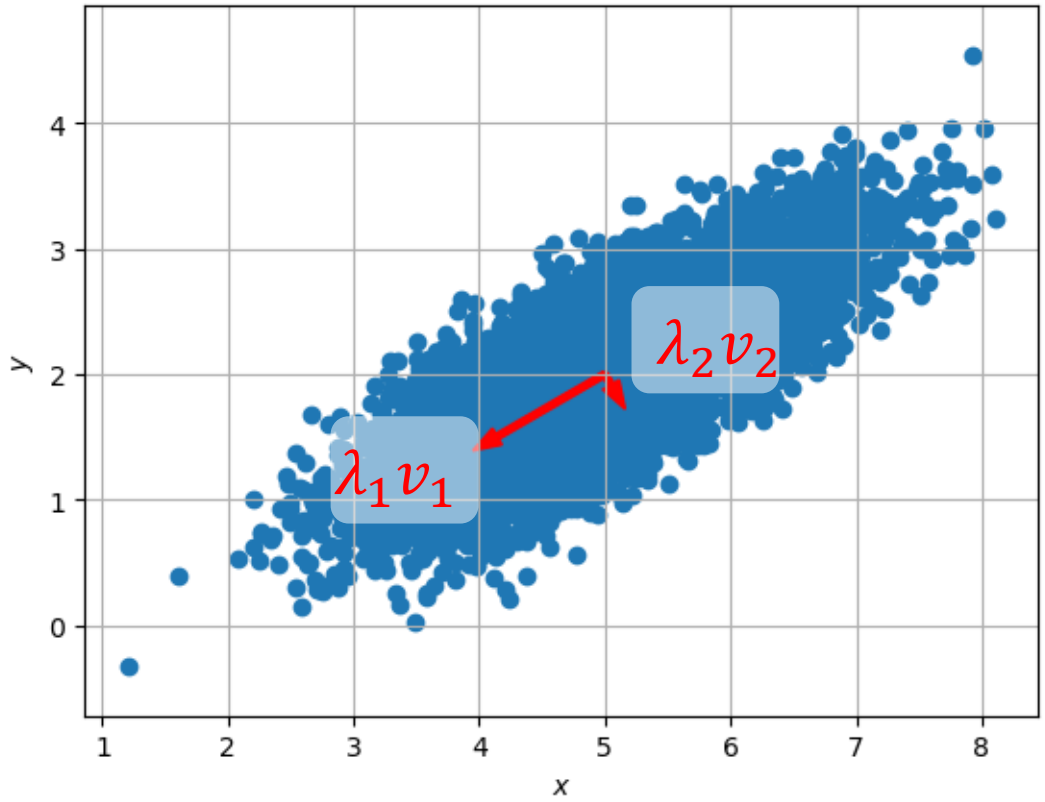
Eigenvectors and Eigenvalues:

$$v_1 = \begin{bmatrix} -0.86 \\ 0.50 \end{bmatrix} \text{ with } \lambda_1 = 0.99$$

$$v_2 = \begin{bmatrix} 0.50 \\ -0.86 \end{bmatrix} \text{ with } \lambda_2 = 0.10$$

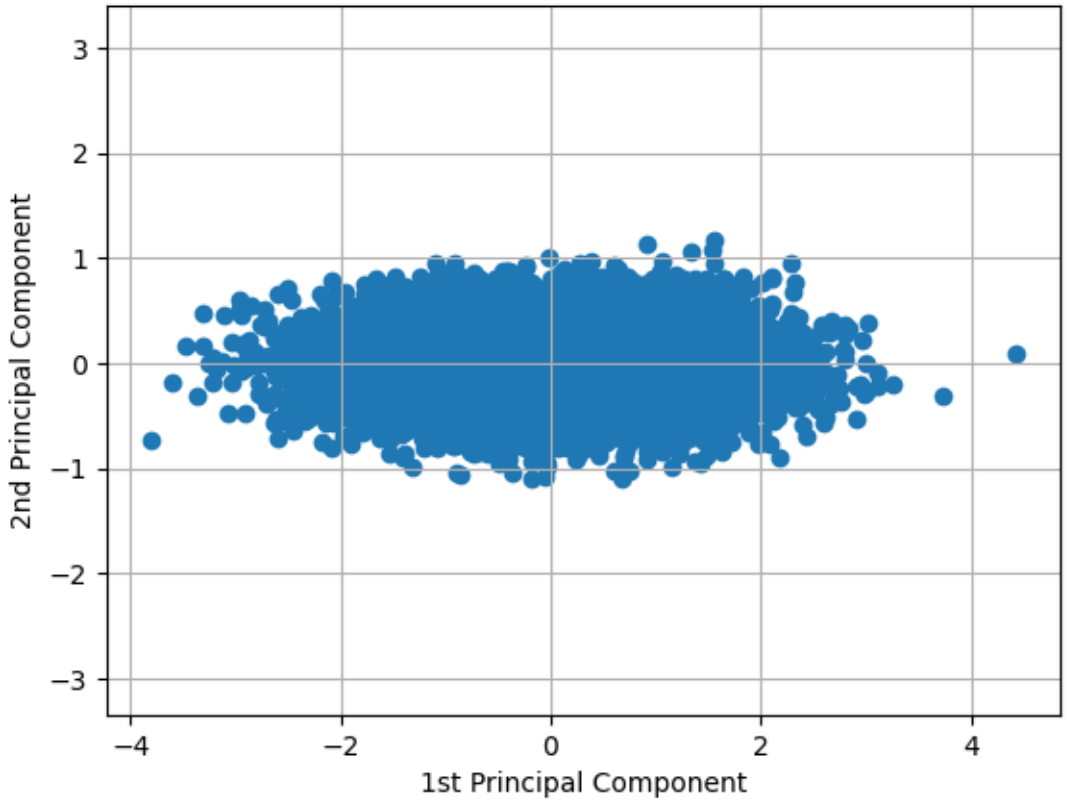
Simple Examples

$$\mathbf{X} = \begin{bmatrix} x_1 & x_2 & \cdots & x_N \\ y_1 & y_2 & \cdots & y_N \end{bmatrix}$$

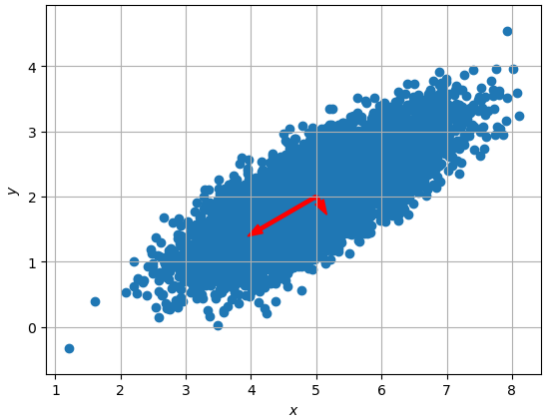


PCA Space

$$\mathbf{Y} = \mathbf{W}^T(\mathbf{X} - \bar{\mathbf{X}}) = \begin{bmatrix} v_1^T \\ v_2^T \end{bmatrix}^T (\mathbf{X} - \bar{\mathbf{X}})$$

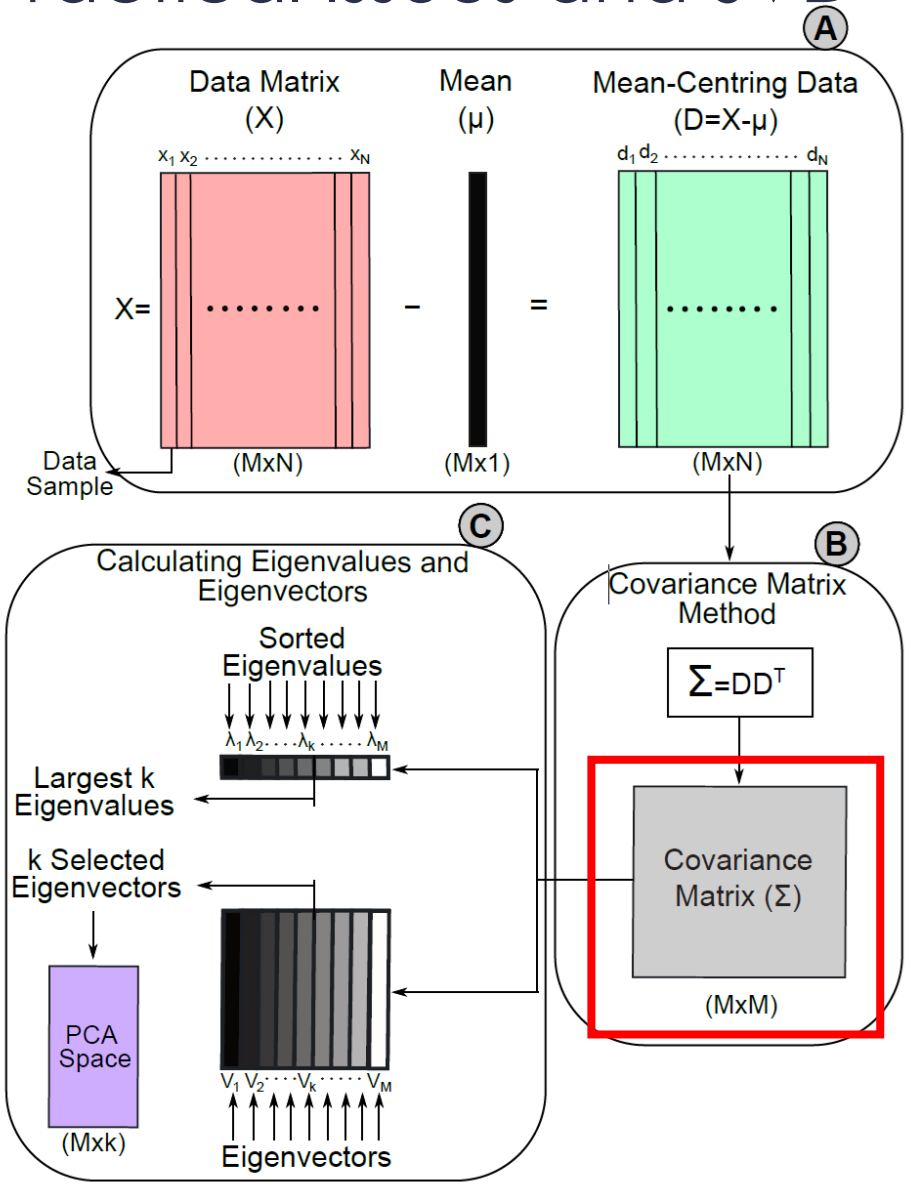


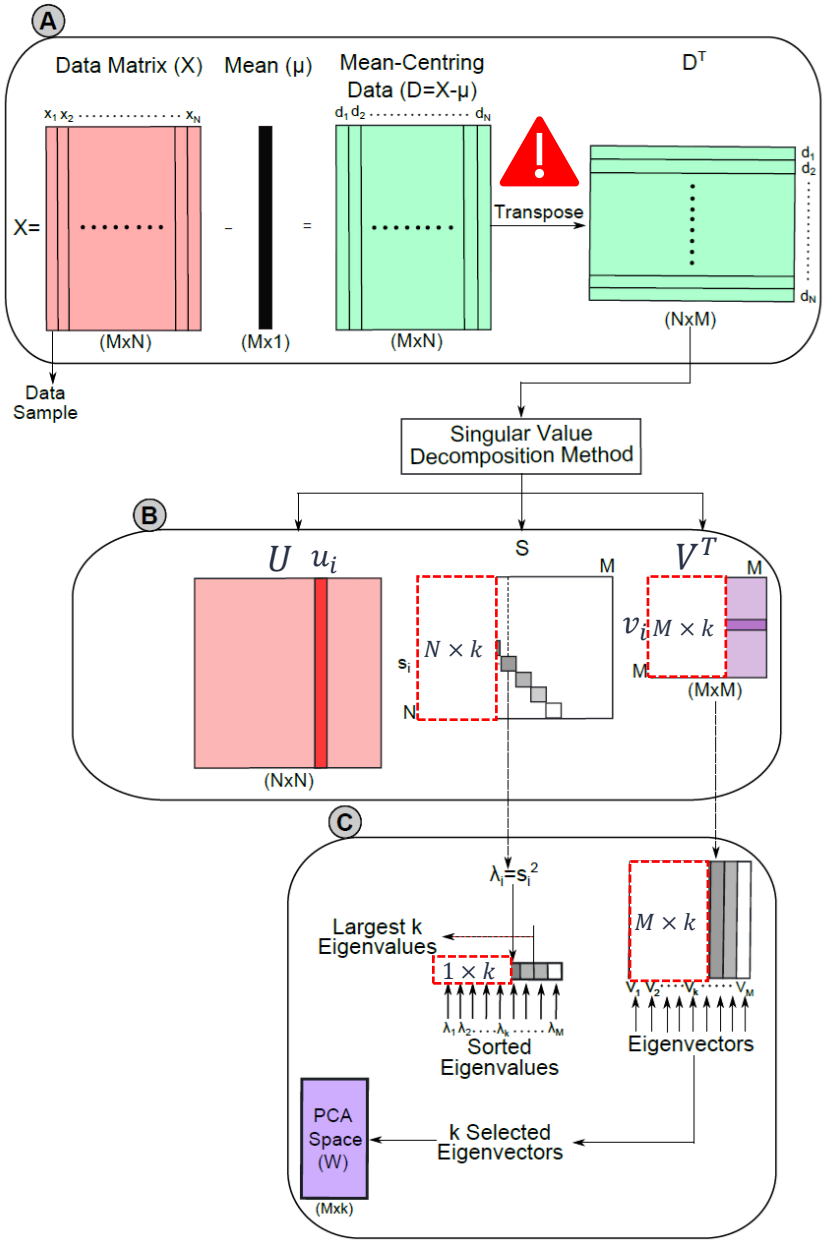
Properties of PCA



- **Intuitive explanation** [Wikipedia]:
  - Fitting an *M*-dimensional ellipsoid to the data (***X***), where each axis of the ellipsoid represents a principal axis (v_i).
  - If some axis of the ellipsoid is small, then the variance (λ_i) along that axis is also small.
- **Linear** transformation of data to new coordinate system (PCA space)
  - **Orthogonal** (lower-dimensional) system. i.e., the principal axes are **perpendicular** and **normalized**
  - The greatest variance of the data comes to lie on the first coordinate (1st PC), the second greatest variance on the second coordinate (2nd PC), etc.
- **PCA assumes** that the original data follows a **Gaussian** distribution.

Practical Issues and SVD





A. Tharwat, "Principal component analysis - a tutorial," *IJAPR*, vol. 3, no. 3, p. 197, 2016, doi: 10.1504/IJAPR.2016.079733.

Code Examples

```
function [PC, PA, Lambda] = pca_via_covariance(X)
%% PCA_via_covariance: Perform PCA using covariance.
%       X - M x N matrix of input data
%           (M dimensions, N trials)
%      PC - M x N matrix of projected data
%           (e.g., the principal components)
%      PA - each column is a principal axis
% Lambdas - M x 1 matrix of variances
    [M, N] = size(X);

    % subtract off the mean for each dimension
    mn = mean(X,2);
    X = X - repmat(mn,1,N);

    % calculate the covariance matrix
    covariance = 1 / (N-1) * X * X';

    % find the eigenvectors and eigenvalues
    [PA, Lambdas] = eig(covariance);

    % extract diagonal of matrix as vector
    Lambdas = diag(Lambdas);
    % sort the variances in decreasing order
    [~, rindices] = sort(-1*Lambdas);
    Lambdas = Lambdas(rindices);
    PA = PA(:, rindices);

    % project original data -> principal comps.
    PC = PA' * X;
end
```

```
function [PC, PA, Lambda] = pca_via_svd(X)
%% PCA_via_SVD: Perform PCA using SVD.
%       X - M x N matrix of input data
%           (M dimensions, N trials)
%      PC - M x N matrix of projected data
%           (e.g., the principal components)
%      PA - each column is a principal axis
% Lambdas - M x 1 matrix of variances
    [M, N] = size(data);

    % subtract off the mean for each dimension
    mn = mean(X,2);
    data = data - repmat(mn,1,N);

    % construct the matrix Y
    Y = data' / sqrt(N-1);

    % SVD does it all
    [u, S, PA] = svd(Y);

    % calculate the variances
    S = diag(S);
    Lambdas = S .* S;


    % project original data -> principal comps.
    PC = PA' * X;
end
```


J. Shlens, "A tutorial on principal component analysis," *arXiv preprint arXiv:1404.1100*, 2014, Accessed: Apr. 18, 2017. [Online]. Available: https://arxiv.org/abs/1404.1100

MATLAB®

MathWorks® Products Solutions Academia Support Community Events Get MATLAB

Help Center | Search Help Center | Help Cen

CONTENTS

« Documentation Home

« AI, Data Science, and Statistics

« Statistics and Machine Learning Toolbox

« Dimensionality Reduction and Feature Extraction

pca

ON THIS PAGE

Syntax
Description
Examples
Input Arguments
Output Arguments
More About
Algorithms
Alternative Functionality
References
Extended Capabilities
Version History
See Also

Documentation More Videos Answers Trial Software Proc

pca

Principal component analysis of raw data

Syntax

```
coeff = pca(X)
coeff = pca(X,Name,Value)
[coeff,score,latent] = pca( __ )
[coeff,score,latent,tsquared] = pca( __ )
[coeff,score,latent,tsquared,explained,mu] = pca( __ )
```

Description

`coeff = pca(X)` returns the principal component coefficients, also known as loadings, for the *n*-by-*p* data matrix `X`. Rows of `X` correspond to observations and columns correspond to variables. The coefficient matrix is *p*-by-*p*. Each column of `coeff` contains coefficients for one principal component, and the columns are in descending order of component variance. By default, `pca` centers the data and uses the singular value decomposition (SVD) algorithm.

`coeff = pca(X,Name,Value)` returns any of the output arguments in the previous syntaxes using additional options for computation and handling of special data types, specified by one or more `Name,Value` pair arguments.

For example, you can specify the number of principal components `pca` returns or an algorithm other than SVD to use.

`[coeff,score,latent] = pca( __ )` also returns the principal component scores in `score` and the principal component variances in `latent`. You can use any of the input arguments in the previous syntaxes.

Principal component scores are the representations of `X` in the principal component space. Rows of `score` correspond to observations, and columns correspond to components.

The principal component variances are the eigenvalues of the covariance matrix of `X`.

`[coeff,score,latent,tsquared] = pca( __ )` also returns the Hotelling's T-squared statistic for each observation in `X`.

`[coeff,score,latent,tsquared,explained,mu] = pca( __ )` also returns `explained`, the percentage of the total variance explained by each principal component and `mu`, the estimated mean of each variable in `X`.

Principal component scores → principal components

python™

scikit learn Install User Guide API Examples Community More Go

Prev Up Next

scikit-learn 1.1.3
Other versions

Please cite us if you use the software.

sklearn.decomposition.PCA
Examples using
sklearn.decomposition.PCA

sklearn.decomposition.PCA

class `sklearn.decomposition.PCA`(*n_components=None, *, copy=True, whiten=False, svd_solver='auto', tol=0.0, iterated_power='auto', n_oversamples=10, power_iteration_normalizer='auto', random_state=None*) [source]

Principal component analysis (PCA).

Linear dimensionality reduction using Singular Value Decomposition of the data to project it to a lower dimensional space. The input data is centered but not scaled for each feature before applying the SVD.

It uses the LAPACK implementation of the full SVD or a randomized truncated SVD by the method of Halko et al. 2009, depending on the shape of the input data and the number of components to extract.

It can also use the scipy.sparse.linalg ARPACK implementation of the truncated SVD.

Notice that this class does not support sparse input. See `TruncatedSVD` for an alternative with sparse data.

Attributes:

components_ : *ndarray of shape (n_components, n_features)*
Principal axes in feature space, representing the directions of maximum variance in the data. Equivalently, the right singular vectors of the centered input data, parallel to its eigenvectors. The components are sorted by decreasing `explained_variance_`.

explained_variance_ : *ndarray of shape (n_components,)*
The amount of variance explained by each of the selected components. The variance estimation uses `n_samples - 1` degrees of freedom.

Equal to n_components largest eigenvalues of the covariance matrix of X.

New in version 0.18.

explained_variance_ratio_ : *ndarray of shape (n_components,)*
Percentage of variance explained by each of the selected components.

If `n_components` is not set then all components are stored and the sum of the ratios is equal to 1.0.

singular_values_ : *ndarray of shape (n_components,)*
The singular values corresponding to each of the selected components. The singular values are equal to the 2-norms of the `n_components` variables in the lower-dimensional space.

New in version 0.19.

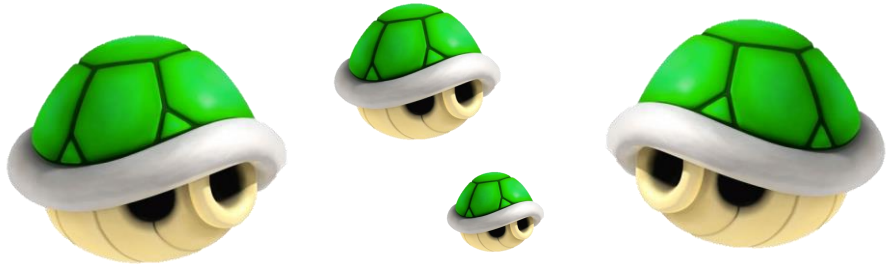
mean_ : *ndarray of shape (n_features,)*
Per-feature empirical mean, estimated from the training set.

Equal to `X.mean(axis=0)`.

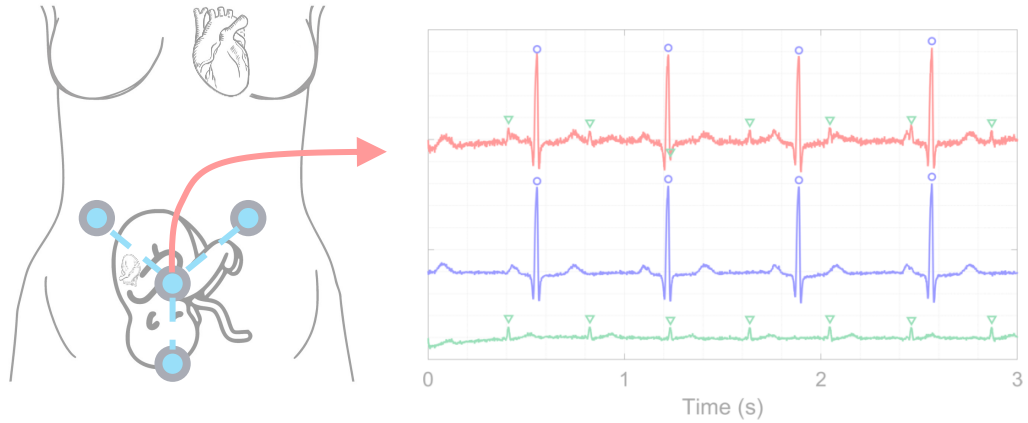
EPFL :: csem

Usage Examples

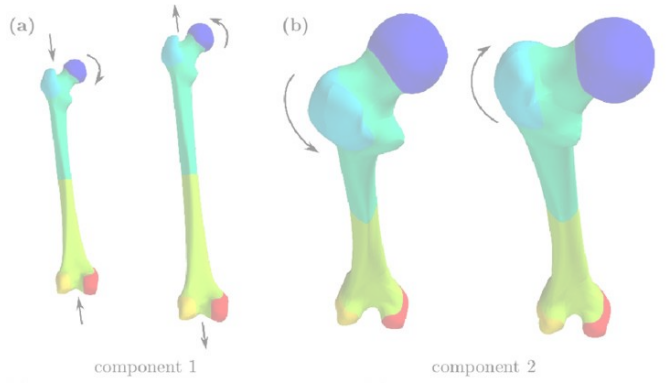
- Dimensionality reduction



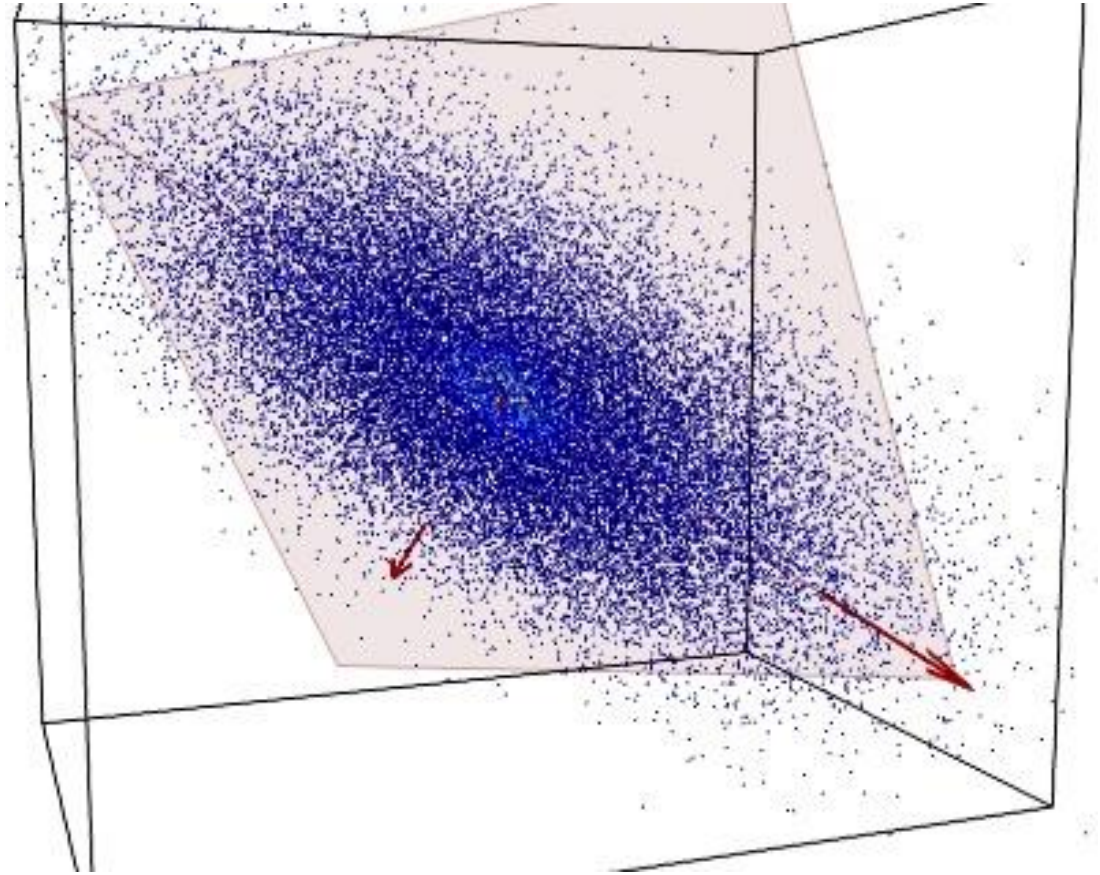
- Blind source separation



- Statistical shape modelling

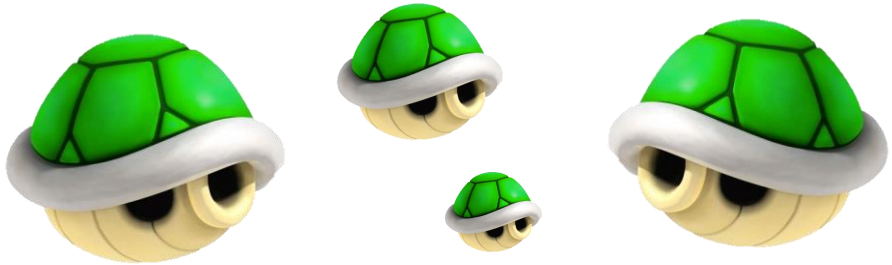


Dimensionality Reduction



https://gael-varoquaux.info/science/ica_vs_pca.html

Dimensionality Reduction – Turtles



- Classical reference
- Jolicoeur and Mosimann studied statistically the size of turtle shells: length (L), width (W), height (H).

Growth, 1960, **24**, 339-354.

SIZE AND SHAPE VARIATION IN THE PAINTED TURTLE.[1]
A PRINCIPAL COMPONENT ANALYSIS

PIERRE JOLICOEUR AND JAMES E. MOSIMANN[2]

Walker Museum, University of Chicago
and
Institut de Biologie, Université de Montréal

TABLE 1
CARAPACE DIMENSIONS OF PAINTED TURTLES (*Chrysemys picta marginata*) IN MM.

24 Males			24 Females		
length	width	height	length	width	height
93	74	37	98	81	38
94	78	35	103	84	38
96	80	35	103	86	42
101	84	39	105	86	40
102	85	38	109	88	44
103	81	37	123	92	50
104	83	39	123	95	46
106	83	39	133	99	51
107	82	38	133	102	51
112	89	40	133	102	51
113	88	40	134	100	48
114	86	40	136	102	49
116	90	43	137	98	51
117	90	41	138	99	51
117	91	41	141	105	53
119	93	41	147	108	57
120	89	40	149	107	55
120	93	44	153	107	56
121	95	42	155	115	63
125	93	45	155	117	60
127	96	45	158	115	62
128	95	45	159	118	63
131	95	46	162	124	61
135	106	47	177	132	67

N = 24:
# turtles

M = 3: # features

P. Jolicoeur and J. E. Mosimann, "Size and shape variation in the painted turtle. A principal component analysis," *Growth*, vol. 24, pp. 339–354, Dec. 1960.

Dimensionality Reduction – Turtles

TABLE 2

MEAN VECTORS $\bar{X}$ AND COVARIANCE MATRICES W

	24 Males			24 Females		
	length	width	height	length	width	height
X	(113.38	88.29	40.71)	(136.00	102.58	51.96)
W	138.77	79.15	37.38	451.39	271.17	168.70
	79.15	50.04	21.65	271.17	171.73	103.29
	37.38	21.65	11.26	168.70	103.29	66.65

TABLE 4

SIZE AND SHAPE VARIATION

	24 Males			24 Females		
Principal axes	1st (major)	2nd (inter-mediate)	3rd (minor)	1st (major)	2nd (inter-mediate)	3rd (minor)
Magnitude of variance	195.28	3.69	1.10	680.40	6.50	2.86
% of total	97.61	1.84	0.55	98.64	0.94	0.41

TABLE 3

COVARIANCE MATRICES Λ AND MATRICES OF DIRECTION COSINES U OF THE PRINCIPAL AXES

	24 Males			24 Females		
Λ	195.28	0.00	0.00	680.40	0.00	0.00
	0.00	3.69	0.00	0.00	6.50	0.00
	0.00	0.00	1.10	0.00	0.00	2.86
U	.84012	.49190	.22854	.81263	.49549	.30676
	—.48811	.86938	—.07696	—.54537	.83213	.10062
	—.23654	—.04690	.97049	—.20540	—.24907	.94645

P. Jolicoeur and J. E. Mosimann, "Size and shape variation in the painted turtle. A principal component analysis," *Growth*, vol. 24, pp. 339–354, Dec. 1960.

μ

Σ

λ_i

W

```python
import pandas as pd
from sklearn.decomposition import PCA
```

```python
# load data
fn = r'1960Jolicoeur_TurtleData.xlsx'
data = pd.read_excel(fn)
data_males = data.iloc[1:, :3]
data_females = data.iloc[1:, 3:]
```

```python
# perform PCA
pca = PCA()
pca.fit_transform(data_females.values);
```

```python
# mean vector X
pca.mean_
```

```
array([136.        , 102.58333333,  51.95833333])
```

```python
# covariance matrix W
pca.get_covariance()
```

```
array([[451.39130435, 271.17391304, 168.69565217],
       [271.17391304, 171.73188406, 103.28623188],
       [168.69565217, 103.28623188,  66.65036232]])
```

```python
# explained variance (eigenvalues lambda)
pca.explained_variance_
```

```
array([680.41230604,   6.49962394,   2.86162075])
```

```python
# percentage of total variance
pca.explained_variance_ratio_*100
```

```
array([98.64285247,  0.94228373,  0.4148638 ])
```

```python
# principal axes in feature space, i.e., the eigenvectors
# NOTE: strictly speaking these are NOT the principal components(!)
pca.components_
```

```
array([[ 0.81264378,  0.49549812,  0.3067437 ],
       [ 0.54536962, -0.83213158, -0.10064297],
       [-0.20538271, -0.24907558,  0.94645618]])
```

EPFL :: CSEM

Dimensionality Reduction – Turtles

- 3 Eigenvectors and Eigenvalues:

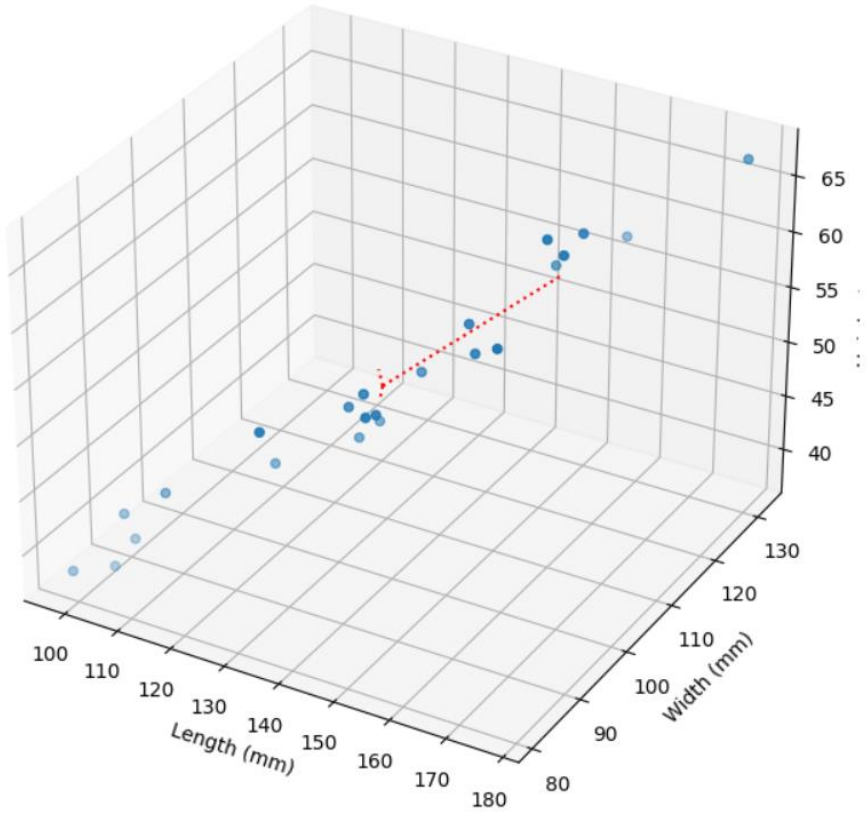
	0.813	-0.545	-0.205	L
	0.496	0.832	-0.249	W
	0.307	0.101	0.947	H
λ	680.4	6.5	2.9	

- First principal component:

$$Y = 0.813\,L + 0.496\,W + 0.307\,H$$

→ explains 98.6% of total variance!

- For a turtle shell, it is not necessary to consider the three features length, width, and height. The abovementioned linear combination is sufficient.
- Other two components indicate marginal relations between:
(2nd) L w.r.t W and H, (3rd) H w.r.t L and W.




```
# explained variance (eigenvalues lambda)
pca.explained_variance_
```

```
array([680.41230604,   6.49962394,   2.86162075])
```

```
# percentage of total variance
pca.explained_variance_ratio_*100
```

```
array([98.64285247,  0.94228373,  0.4148638 ])
```

```
# principal axes in feature space, i.e., the eigenvectors
# NOTE: strictly speaking these are NOT the principal components(!)
pca.components_
```

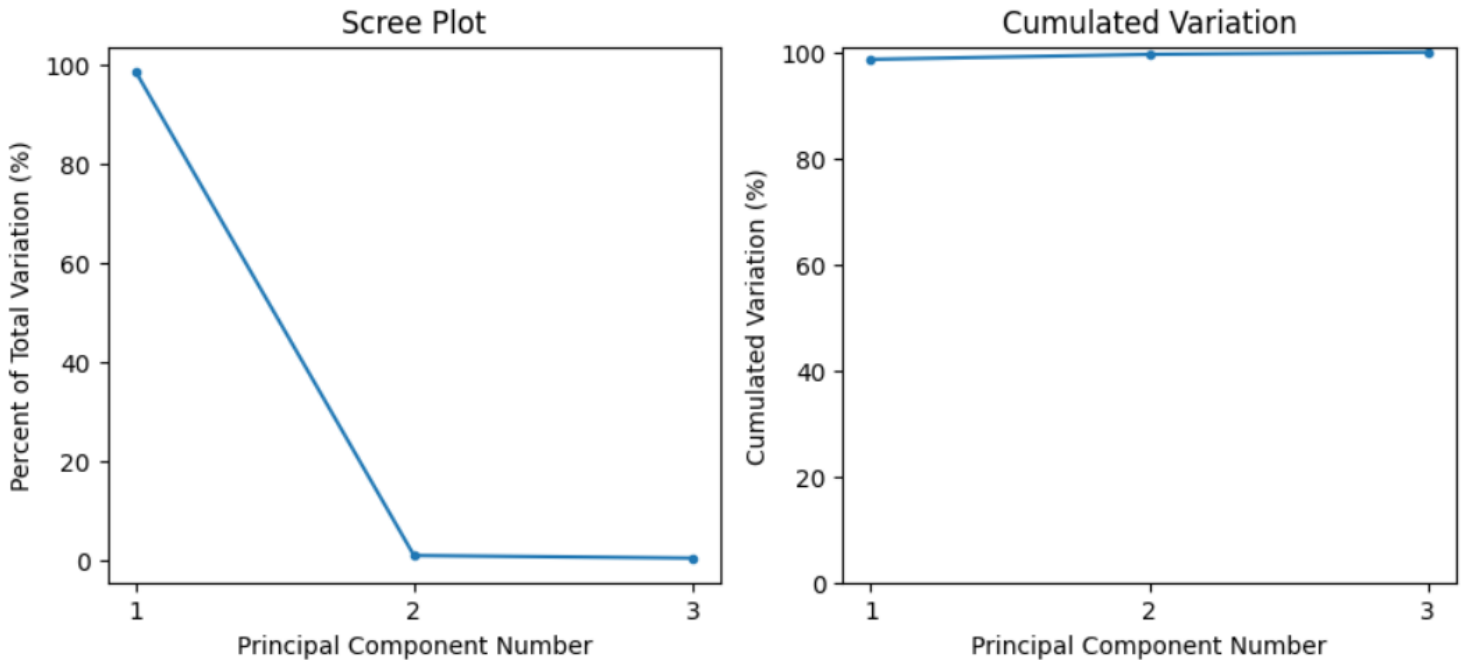
```
array([[ 0.81264378,  0.49549812,  0.3067437 ],
       [ 0.54536962, -0.83213158, -0.10064297],
       [-0.20538271, -0.24907558,  0.94645618]])
```

EPFL :: csem

Dimensionality Reduction – Turtles

```
# scree plot
plt.subplot(121)
plt.plot(np.arange(pca.n_components_) + 1, 100*pca.explained_variance_ratio_, '.-')
plt.xlabel('Principal Component Number');
plt.ylabel('Percent of Total Variation (%)')
plt.title('Scree Plot')

# cumulated variance
plt.subplot(122)
plt.plot(np.arange(pca.n_components_) + 1, np.cumsum(100*pca.explained_variance_ratio_), '.-')
plt.xlabel('Principal Component Number')
plt.ylabel('Cumulated Variation (%)')
plt.title('Cumulated Variation')
plt.show()
```



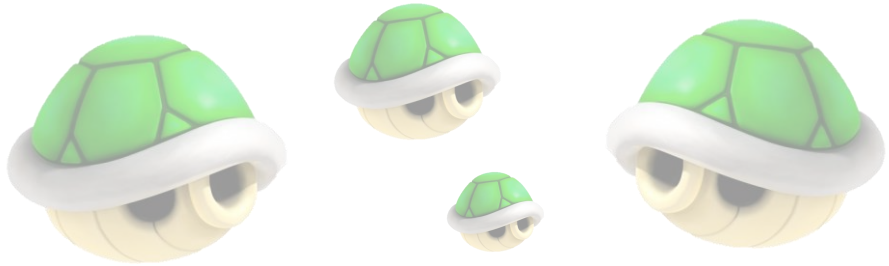
Dimensionality Reduction – Remarks

- It may be that the features have widely different amplitude ranges, which may lead to "numerically null" rows.
  - Use **whitening/normalization:** $\boldsymbol{X}_N = \begin{bmatrix} \frac{1}{\sigma_1} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \frac{1}{\sigma_M} \end{bmatrix} \boldsymbol{X}$

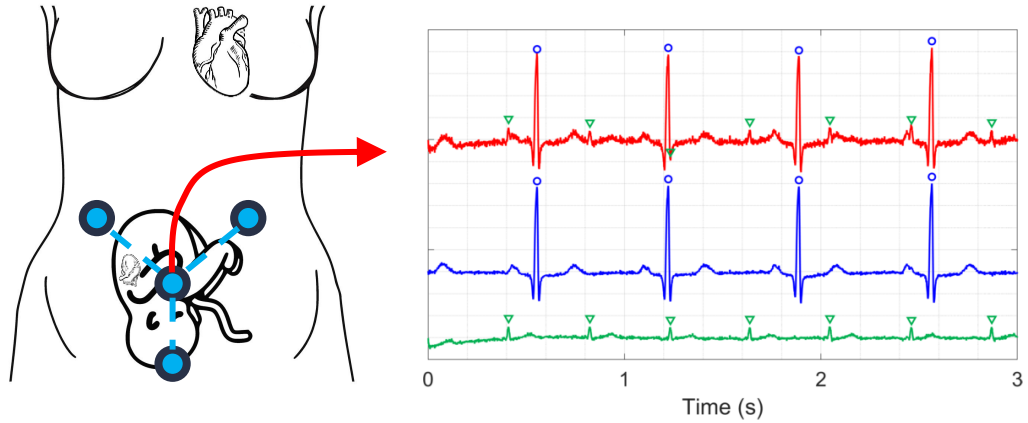
- The PCA works well only for **linear relations** between features.
  - If, for instance, there is a product-type relation, PCA will be a lot less useful.

Usage Examples

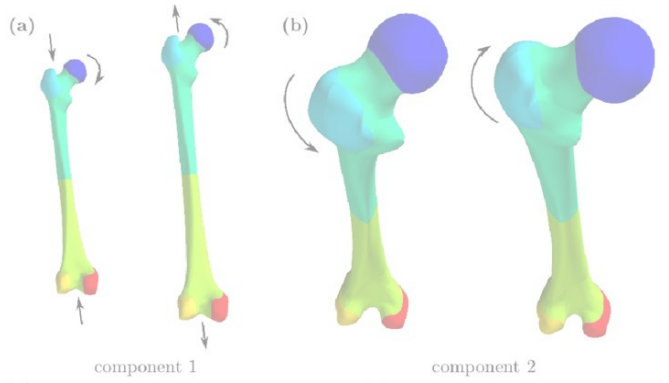
- Dimensionality reduction



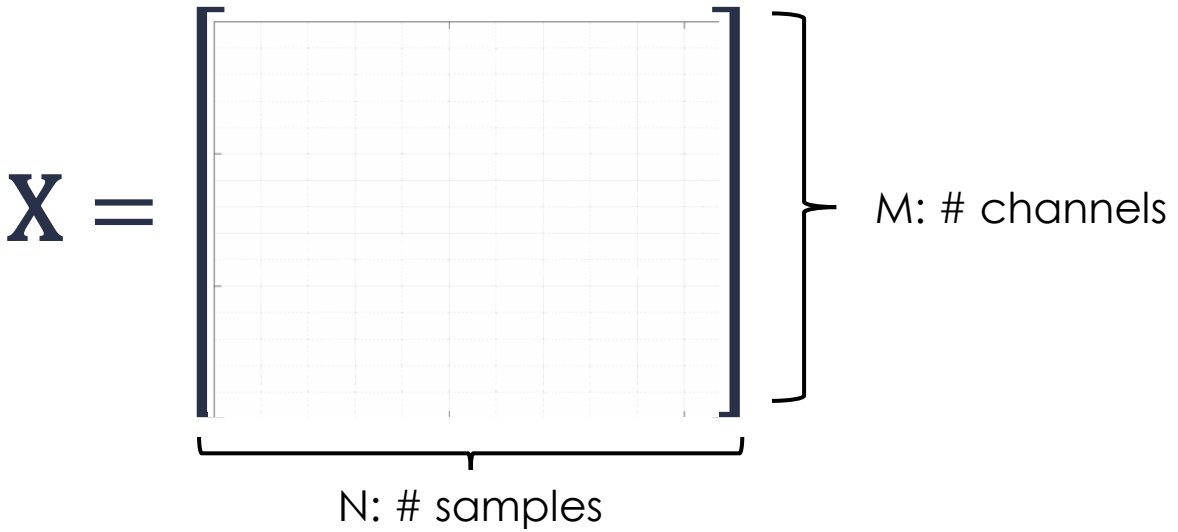
- Blind source separation



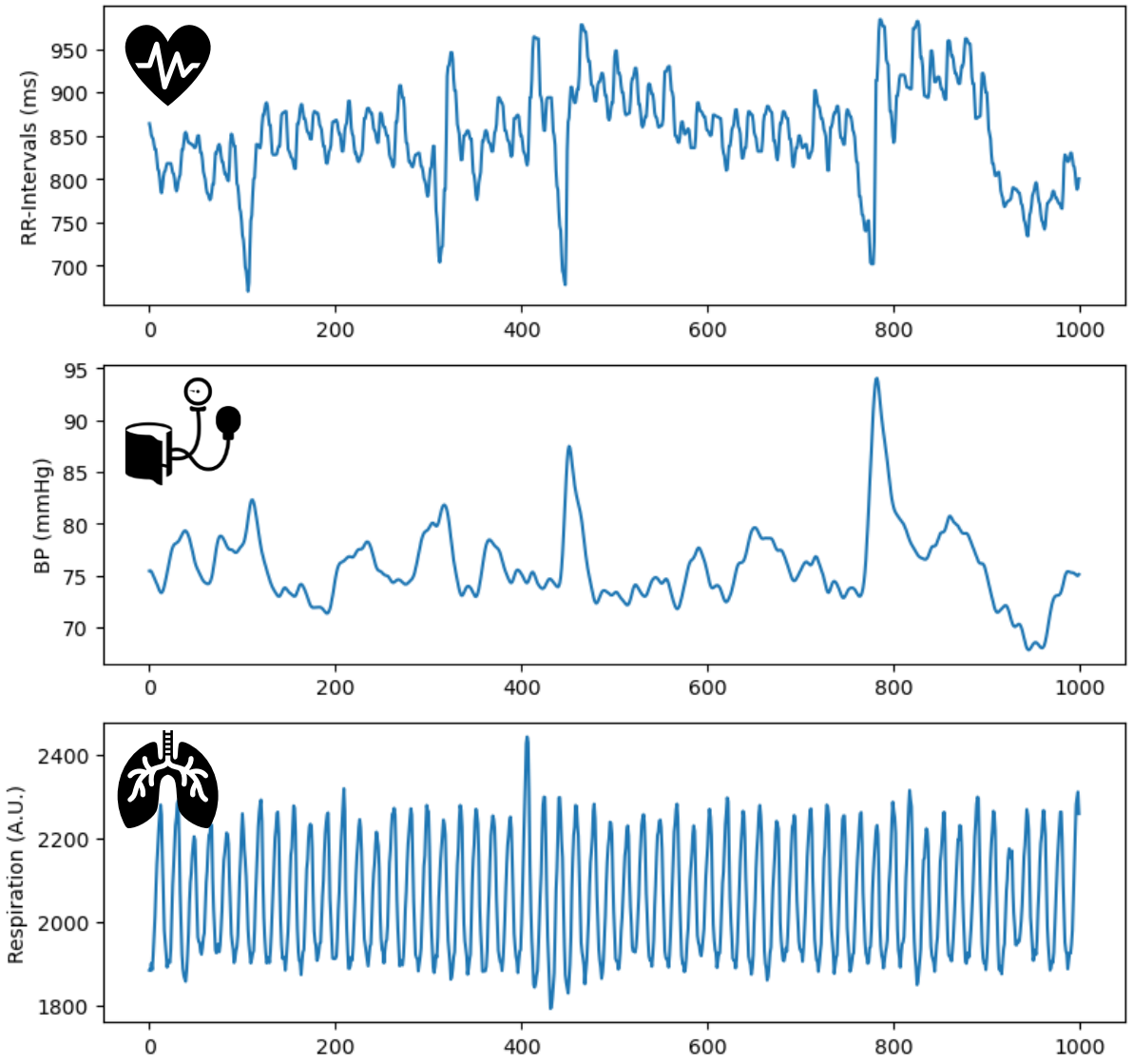
- Statistical shape modelling



PCA and Signals – Blind Source Separation



PCA and Signals – Blind Source Separation



Example:

1. RR intervals
2. Blood pressure
3. Respiration

$$\mathbf{X} = \begin{bmatrix} \\ \\ \\ \end{bmatrix}$$

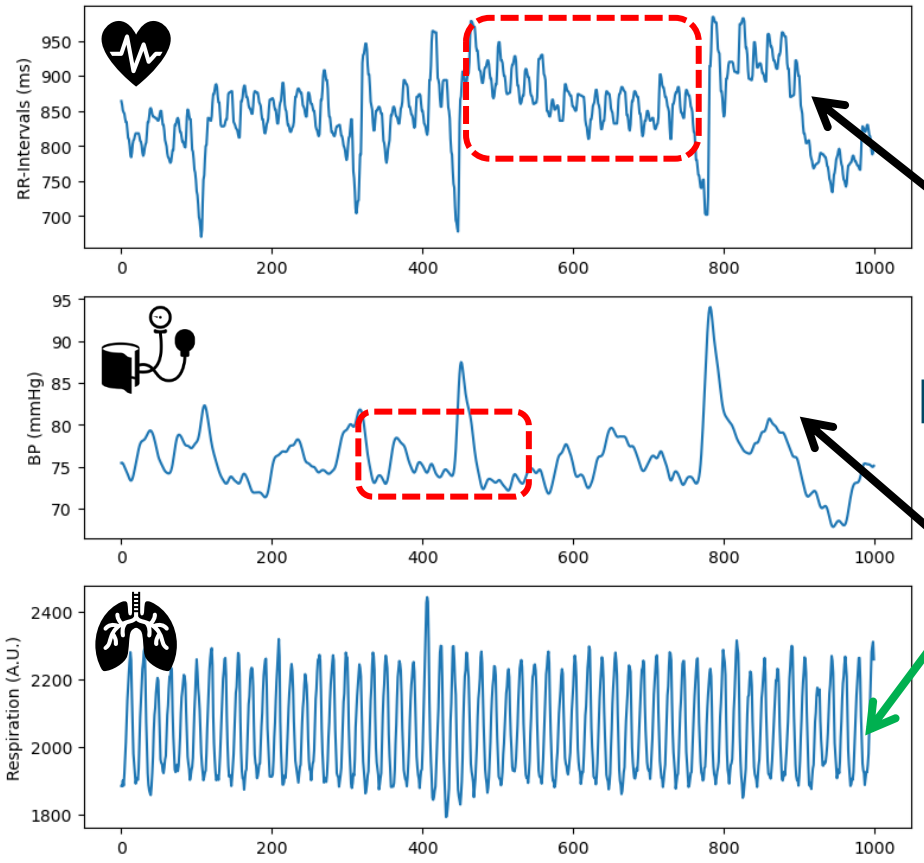
M: 3 channels

N: 1000 samples

EPFL :: csem

PCA and Signals – Blind Source Separation

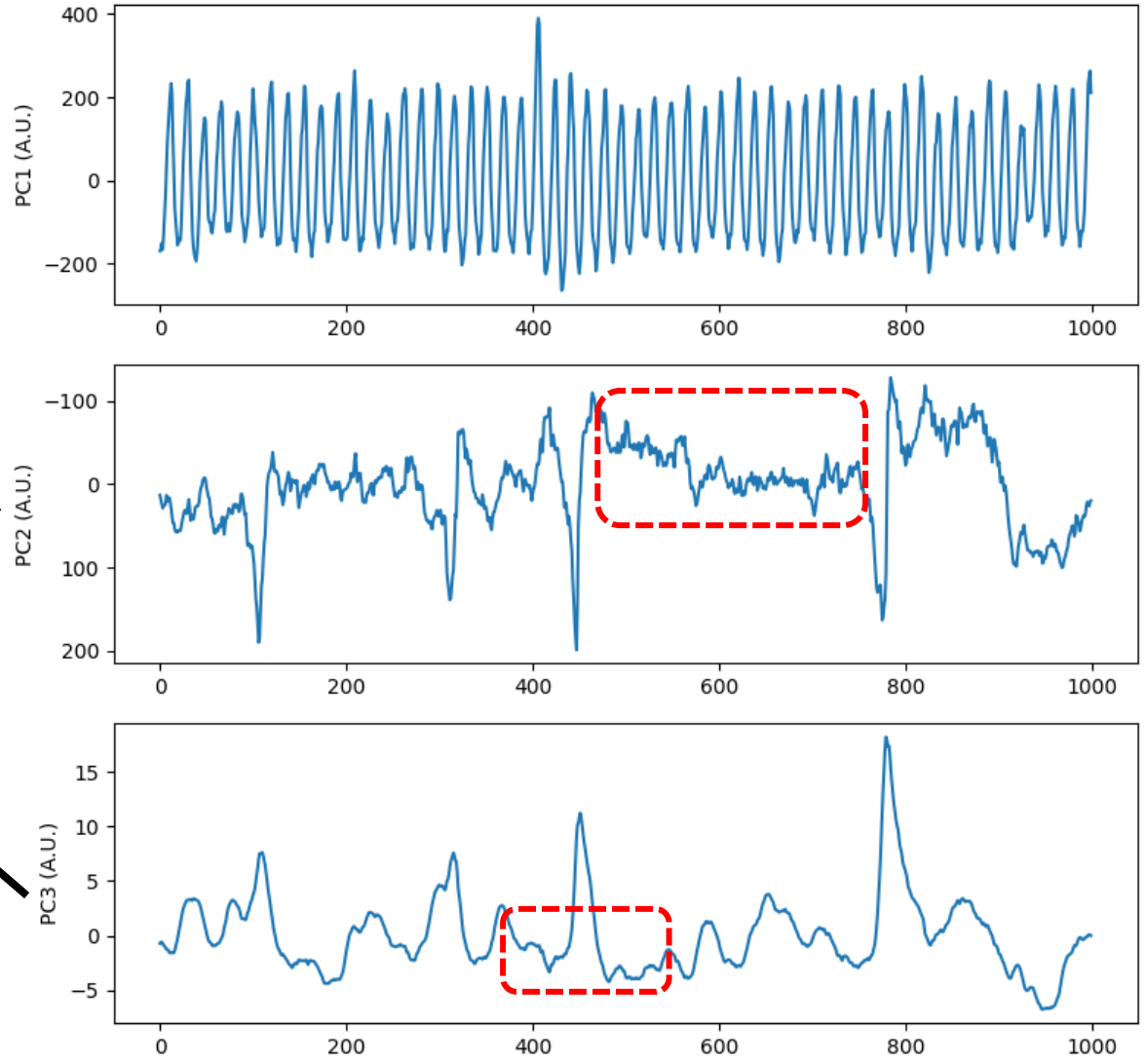
Input Signals



PCA

$\mathbf{Y} = \mathbf{W}^T \boldsymbol{D}$

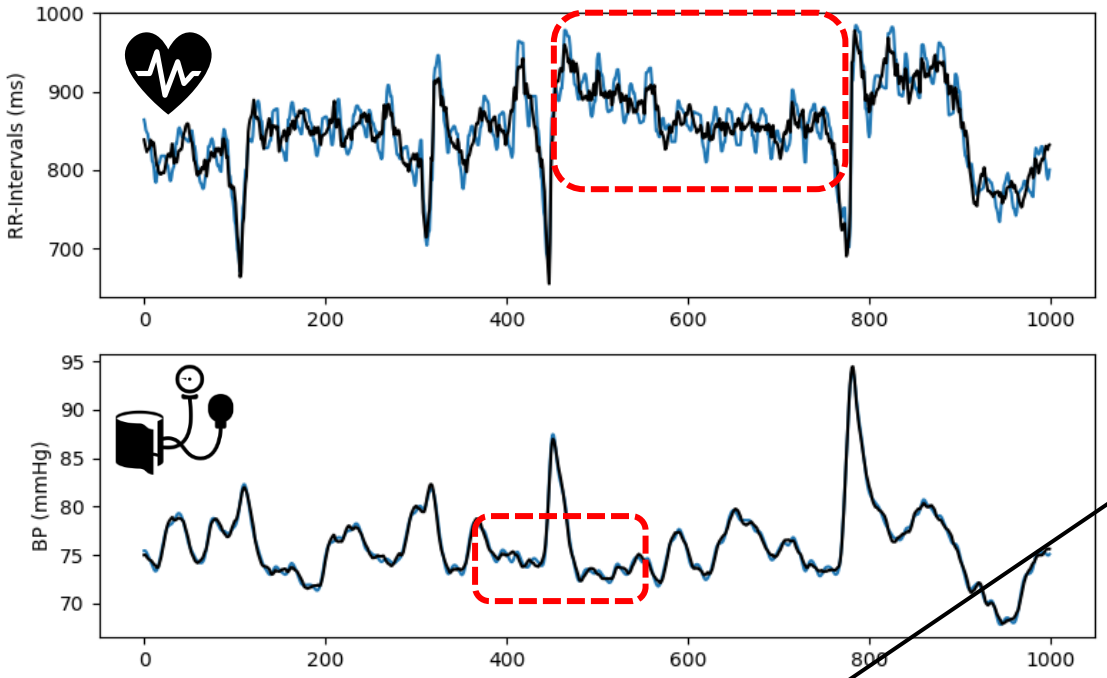
Principal Components



EPFL :: csem

PCA and Signals – Blind Source Separation

Input Signals with Suppressed Respiration



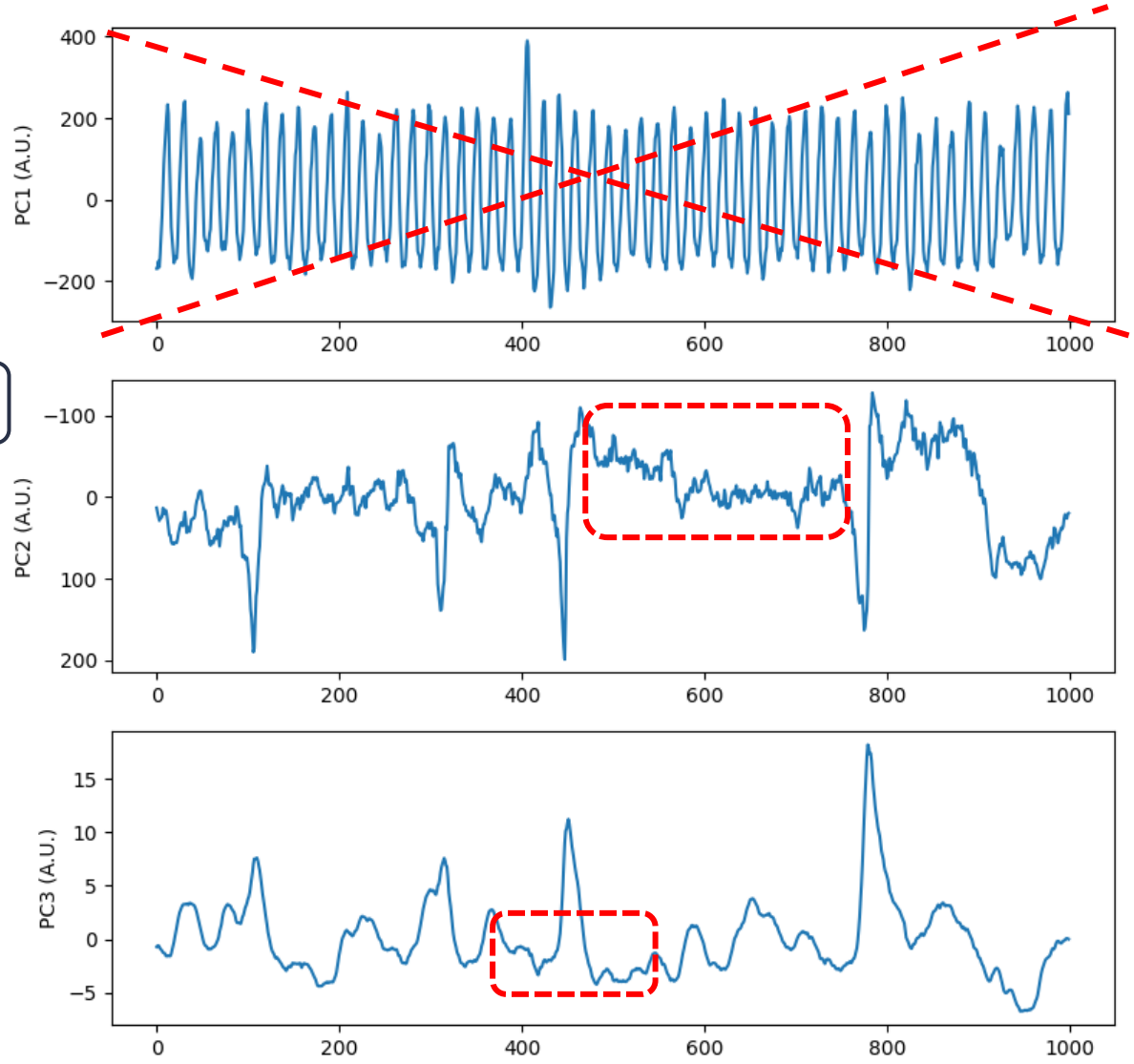
$$\hat{X} = \hat{W}Y$$

```
pca.components_
```

```
array([[-1.50046925e-01, -2.57525767e-03, -9.88675922e-01],
       [-9.88584922e-01, -1.33950730e-02, -1.50068065e-01],
       [-1.36298448e-02,  9.99906966e-01,  5.35971384e-04]])
```

0 0 0

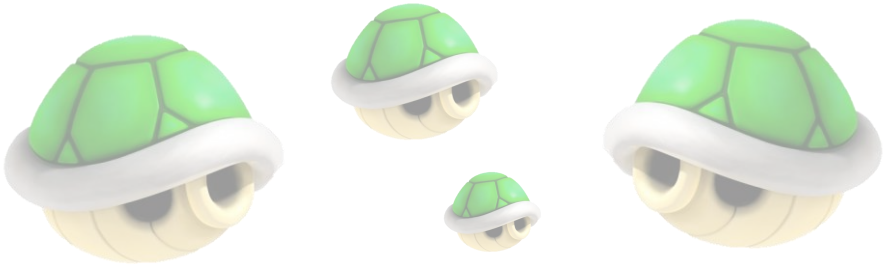
Principal Components



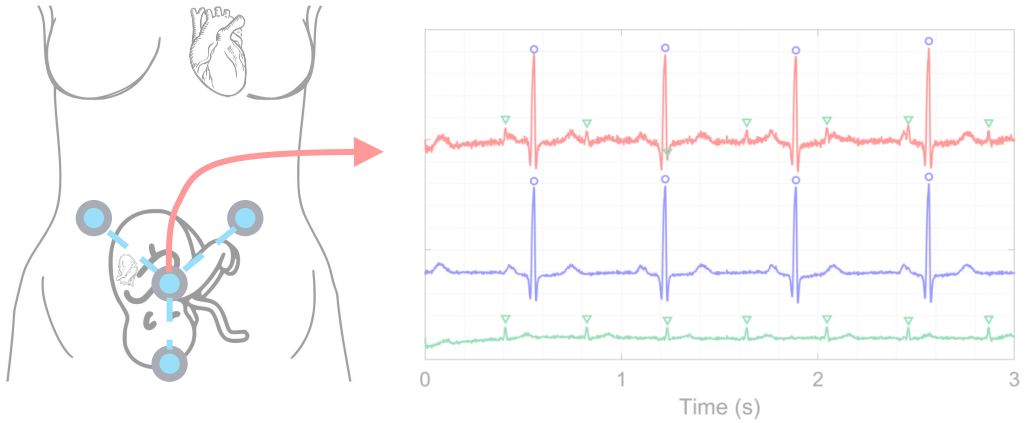
EPFL :: csem

Usage Examples

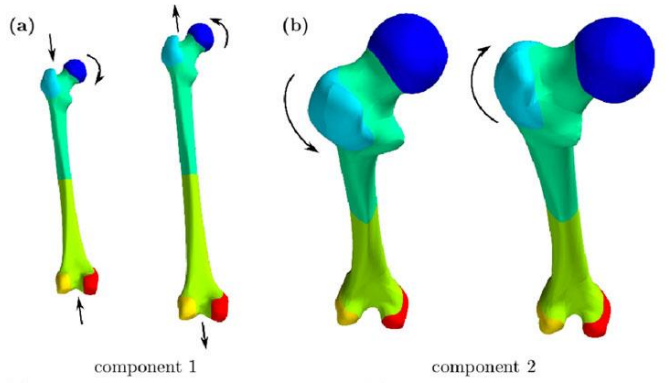
- Dimensionality reduction



- Blind source separation



- Statistical shape modelling



Statistical Shape Modelling – Motivation

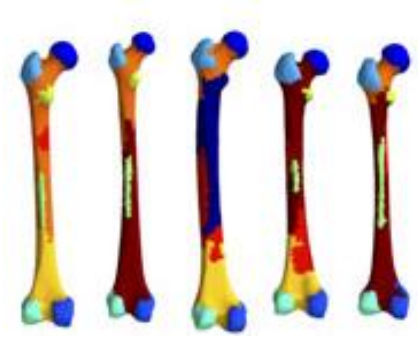
$$\mathbf{X} = \begin{bmatrix} x_1^1 & \cdots & x_1^N \\ y_1^1 & & y_1^N \\ z_1^1 & & z_1^N \\ \vdots & \cdots & \vdots \\ x_S^1 & & x_S^N \\ y_S^1 & & y_S^N \\ z_S^1 & & z_S^N \end{bmatrix}$$

M: 3*S = 1902
with S=634 surface points

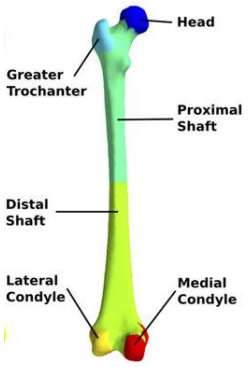
N: 41 femurs

- Mean Shape – knowledge about general shape
- Describe natural anatomical variability (PCA)

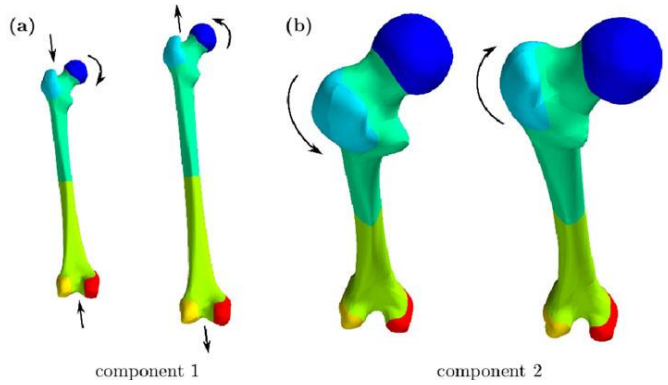
Image Data CT-Scans → 3D Surface Representation → Surface Alignment → PCA → $\begin{bmatrix} v_1 & \cdots & v_M \\ \lambda_1 & \cdots & \lambda_M \end{bmatrix}$



Mean Shape: $\overline{\mathbf{X}}$

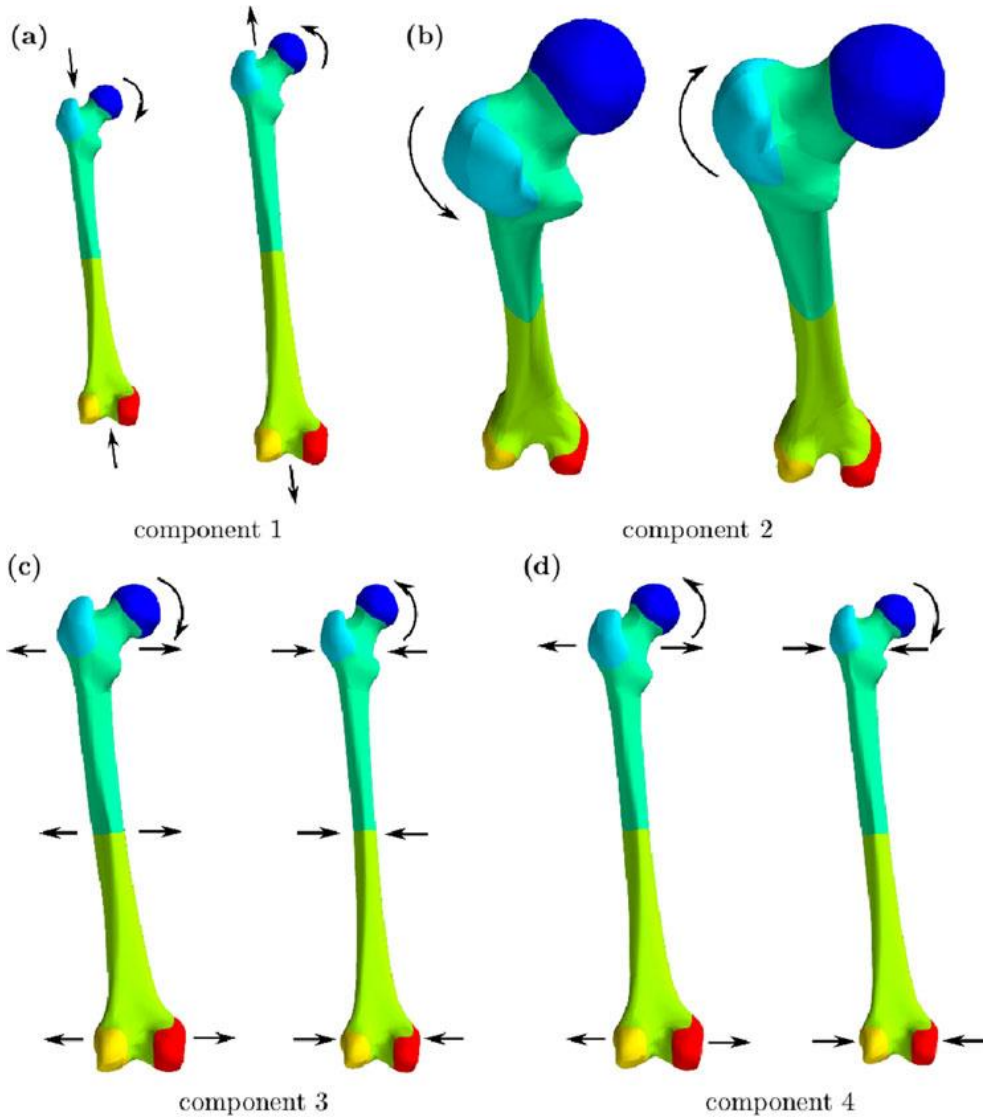


Shape Variation



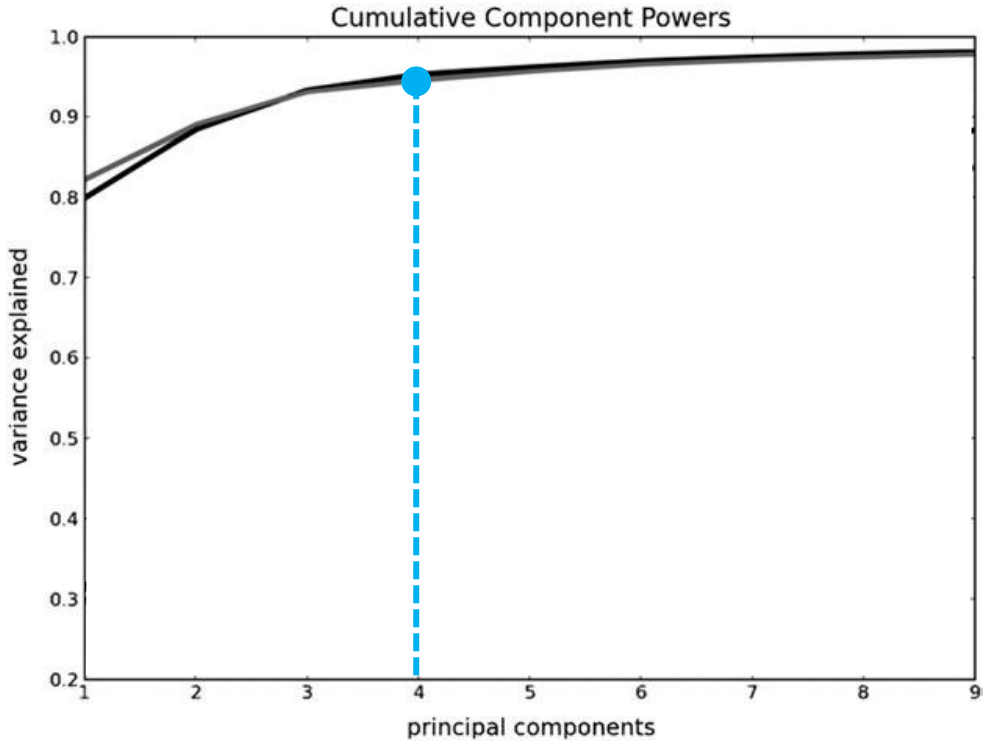
J. Zhang et al., "An anatomical region-based statistical shape model of the human femur" *COMPUT METHOD BIOMEC*, 2:3, 176-185, 2014, doi: 10.1080/21681163.2013.878668.

Statistical Shape Modelling – Femurs



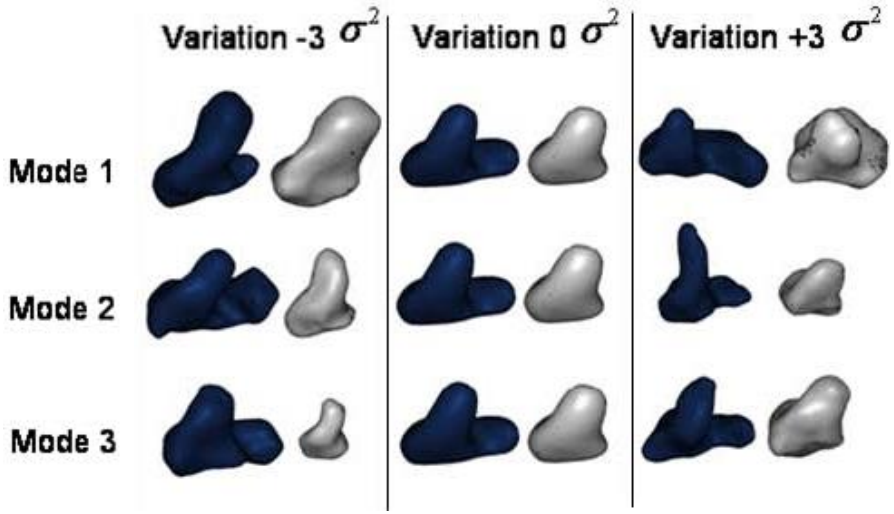
Mean Shape Eigenvectors

$$\hat{\mathbf{X}} = \bar{\mathbf{X}} + \sum_{i}^{4} v_i \cdot b_i, \qquad \text{with } |b_i| \leq 3\lambda_i$$

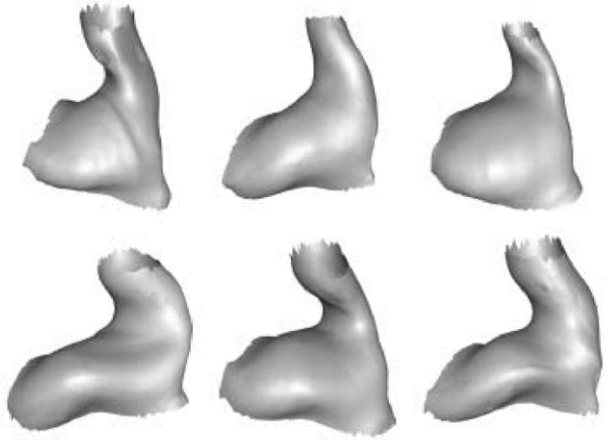


EPFL :: CSEM

Statistical Shape Modelling – Hearing Aids



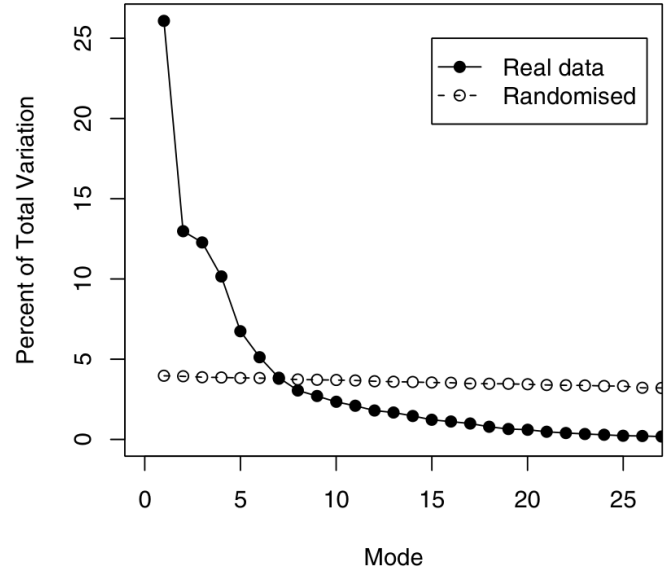
G. Unal, et al., "Customized Design of Hearing Aids Using Statistical Shape Learning," in *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2008*, Berlin, Heidelberg, 2008, vol. 5241, pp. 518–526. doi: 10.1007/978-3-540-85988-8_62.



(a) Mode 1 (b) Mode 2 (c) Mode 3

Fig. 2. Pure shape model. Each shape has been generated by varying the first three modes of variation between −3 (top) and +3 (bottom) standard deviations

Scree Plot



R. Paulsen, et al., "Building and Testing a Statistical Shape Model of the Human Ear Canal," in *Medical Image Computing and Computer-Assisted Intervention — MICCAI 2002*, Berlin, Heidelberg, 2002, vol. 2489, pp. 373–380. doi: 10.1007/3-540-45787-9_47.

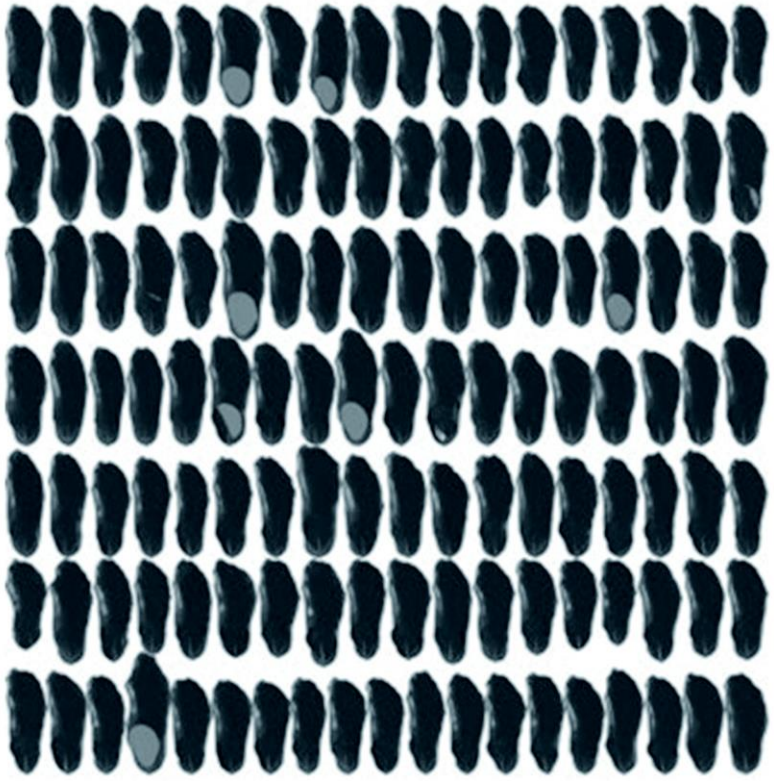
EPFL :: csem

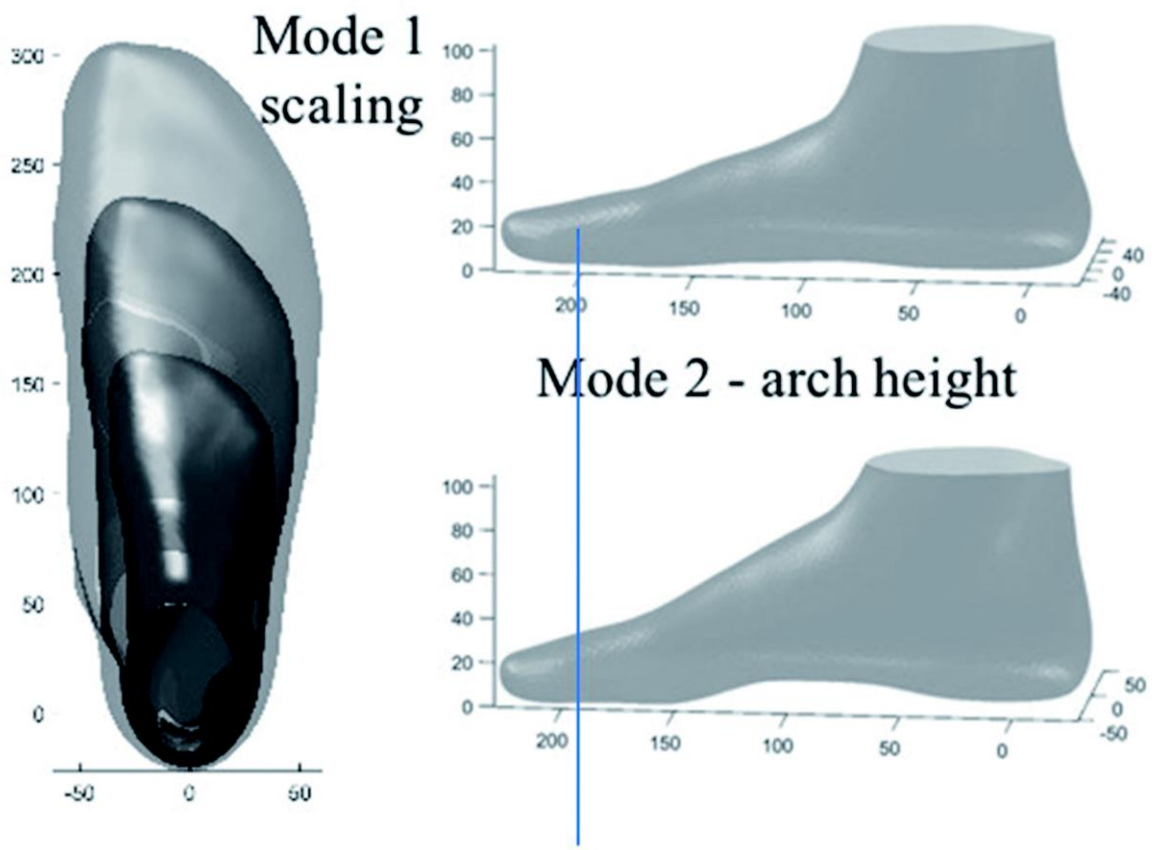
Statistical Shape Modelling – Feet

Statistical shape modelling describes anatomic variation in the foot

Bryan P. Conrad[a*], Michael Amos[b], Irene Sintini[c], Brian Robert Polasek[c] and Peter Laz[c]

[a]Nike Sport Research Lab, Portland, OR, USA; [b]Nike Inc, Nike Sports Research Lab, Beaverton, OR, USA; [c]University of Denver, Denver, CO, USA





B. P. Conrad, et al., "Statistical shape modelling describes anatomic variation in the foot," *Footwear Science*, vol. 11, no. sup1, pp. S203–S205, Jun. 2019, doi: 10.1080/19424280.2019.1606334.

Limitations of PCA

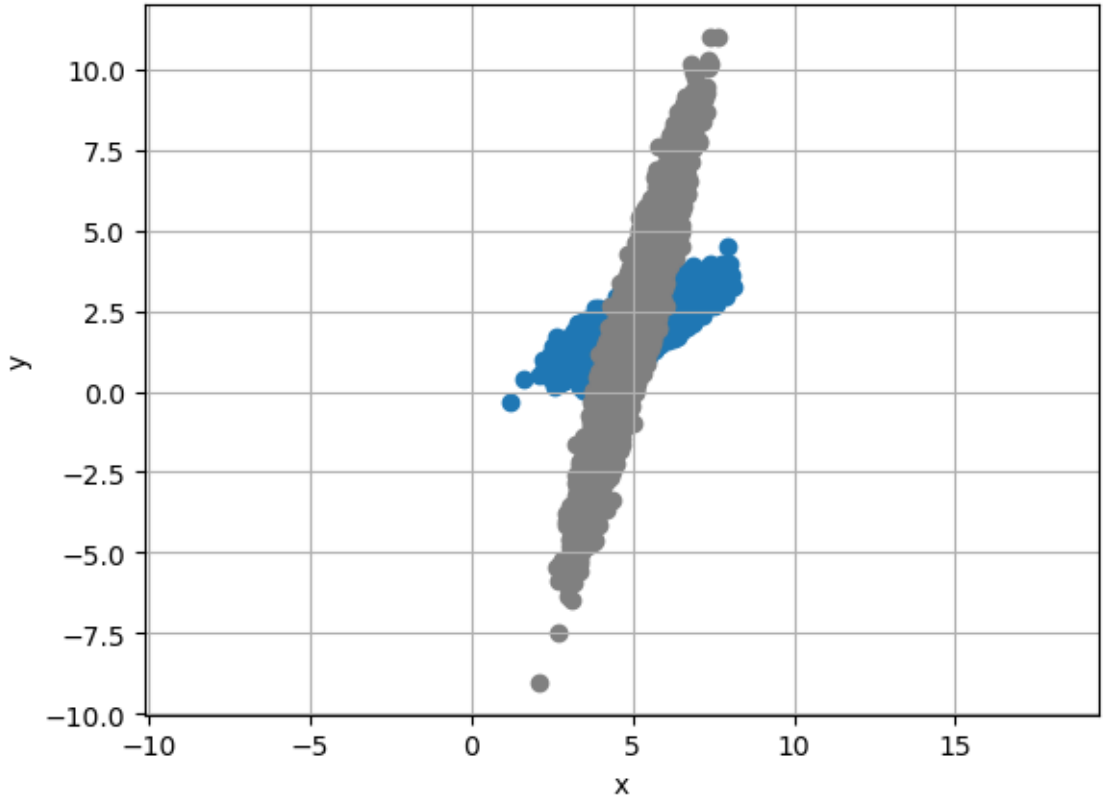
- PCA works well only for linear relations between features.
  - If, for instance, there is a product-type relation, PCA will be a lot less useful.
- PCA is sensitive to outliers
  - Use robust PCA (RPCA)
- PCA assumes that the original data follows a Gaussian distribution

- Try alternatives such as independent component analysis (ICA)

A First Limitation of PCA

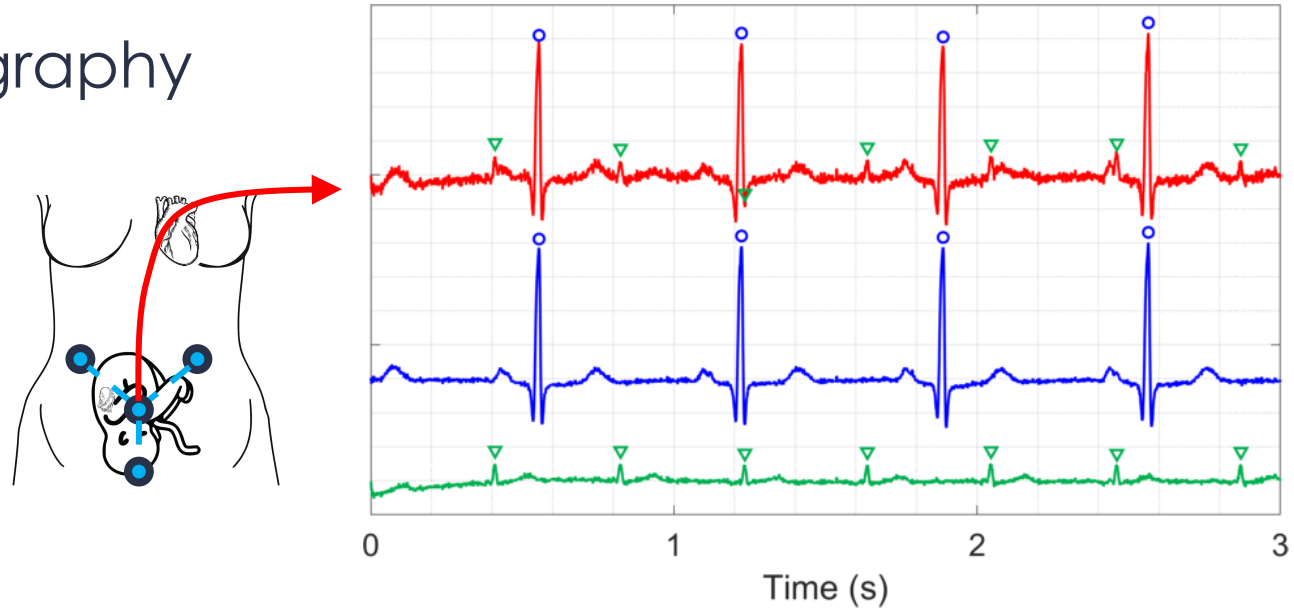
$$\mathbf{X} = \underbrace{\begin{bmatrix} x_1 & x_2 & \cdots & x_N \\ y_1 & y_2 & \cdots & y_N \end{bmatrix}}_{\text{N = 20'000 \# samples}} \Big\} \text{M = 2 \# dimensions}$$

- Two **non**-orthogonal 2D Gaussians

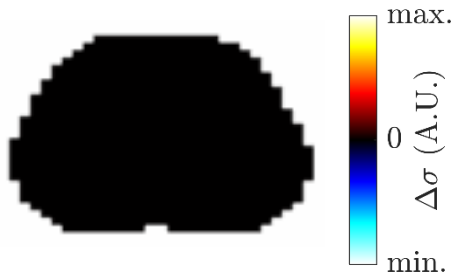


Today's Lab – Blind Source Separation Using PCA

1. fECG: Fetal Electrocardiography

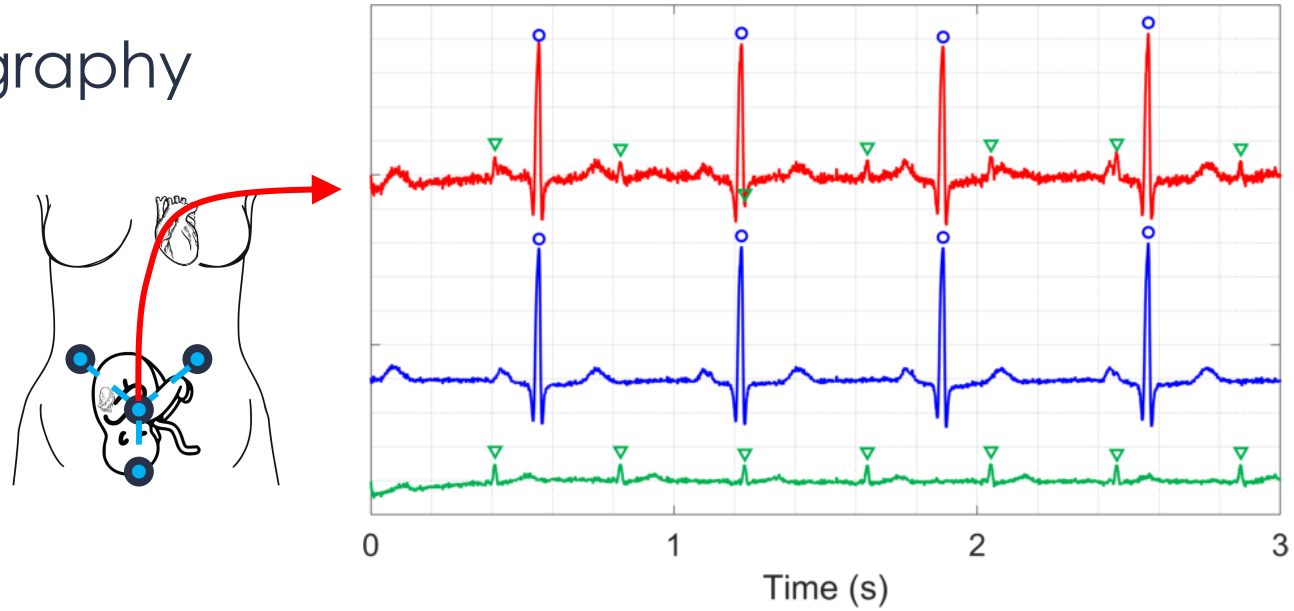


2. EIT: Electrical Impedance Tomography

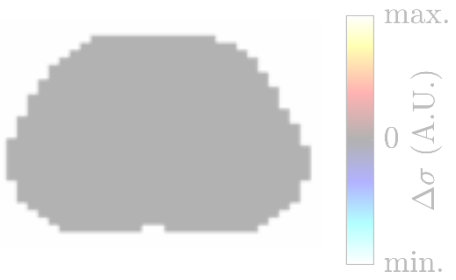


Lab #1

1. fECG: Fetal Electrocardiography



2. EIT: Electrical Impedance Tomography

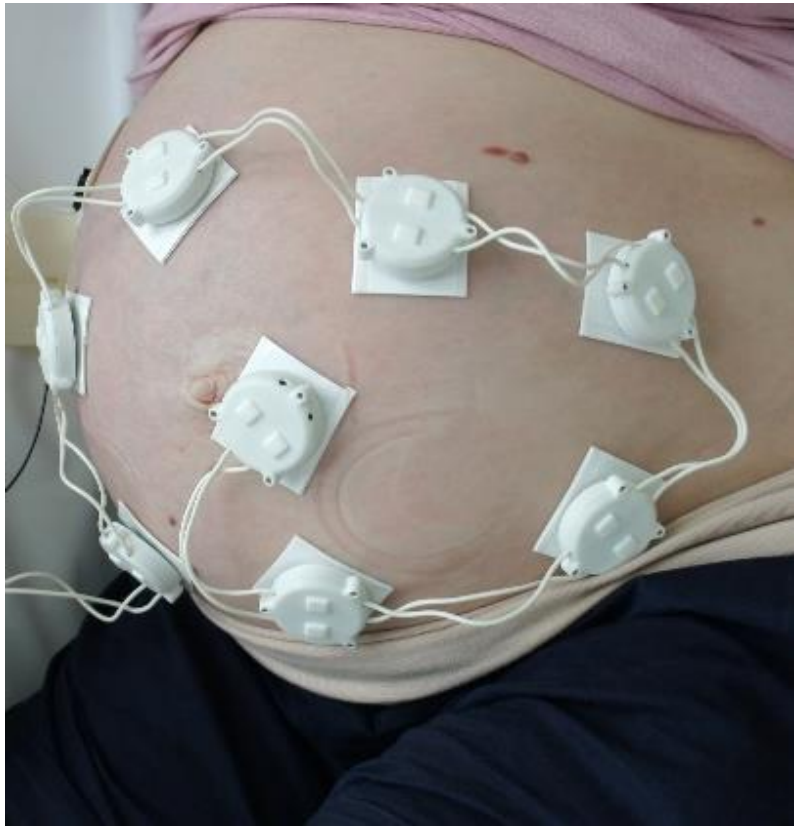


Abdominal ECG for Fetal Heart Rate Estimation

CTG: Cardiotocography

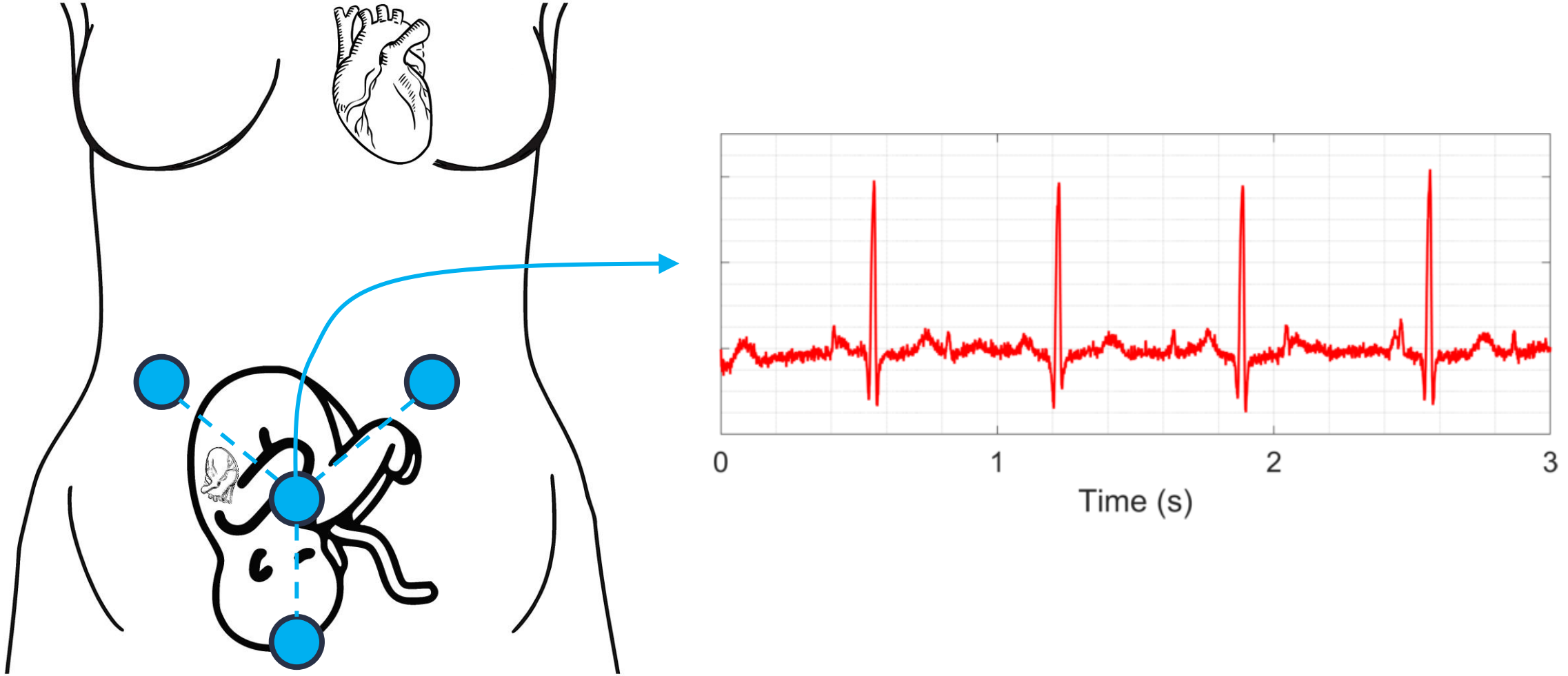
Source: Philips CTG FM30

aECG: Abdominal ECG

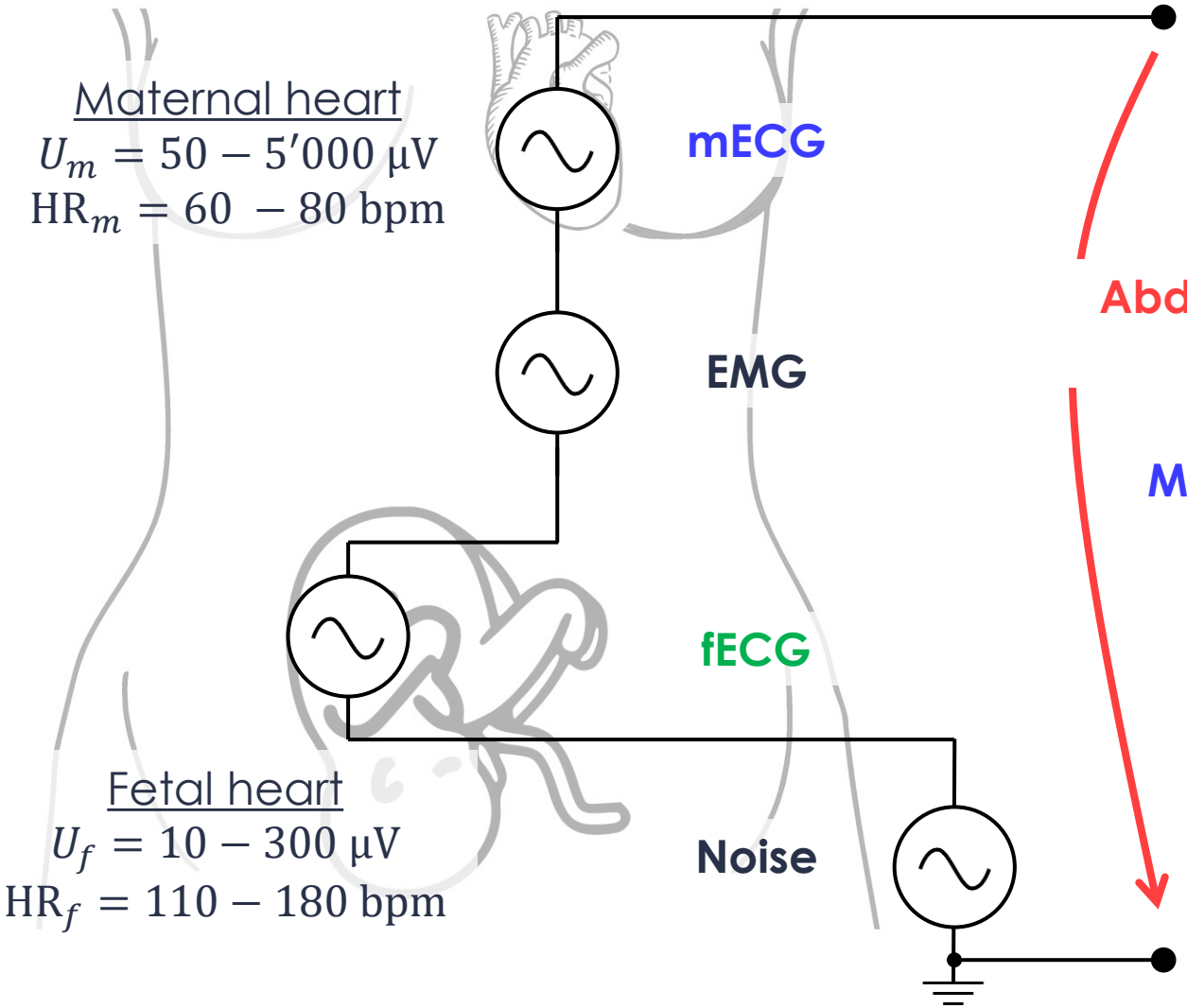


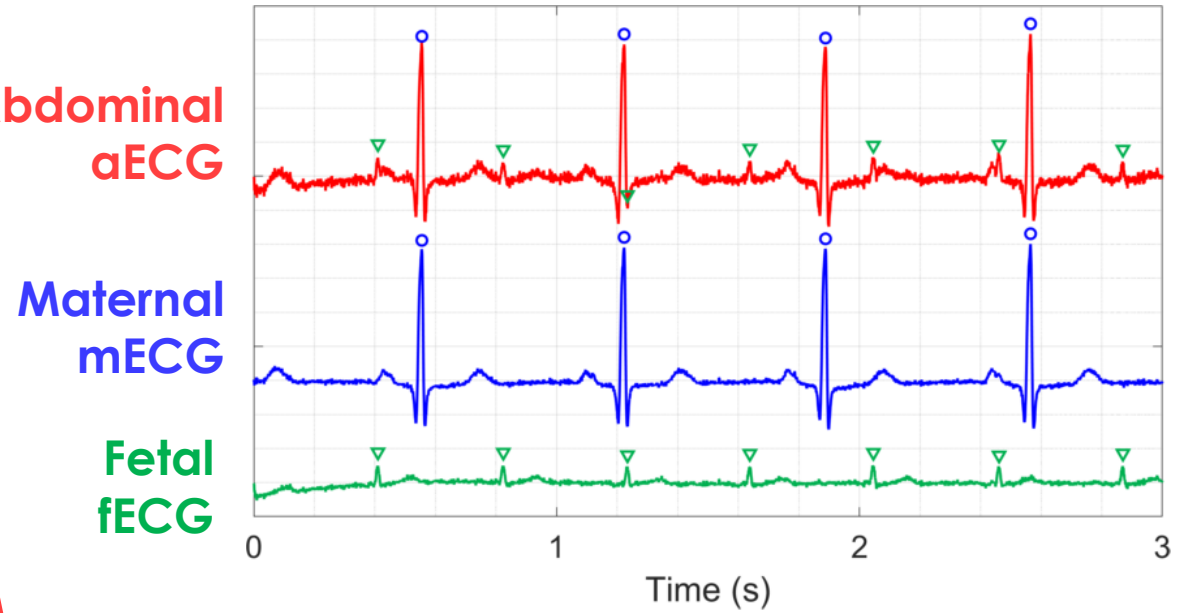
Source: CSEM

Fetal Physiology and Abdominal ECG



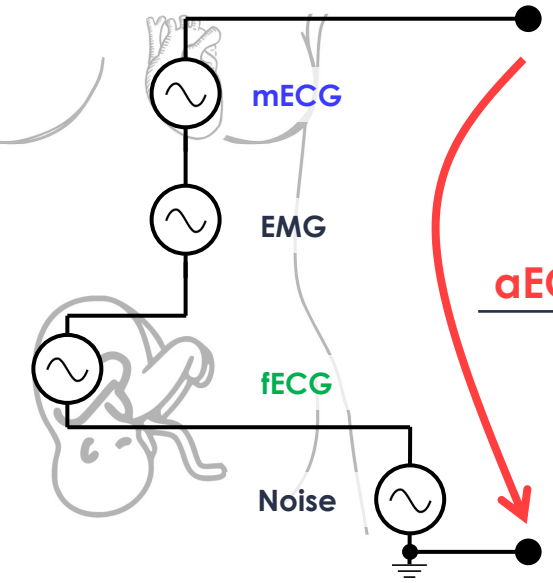
Challenges of Fetal ECG

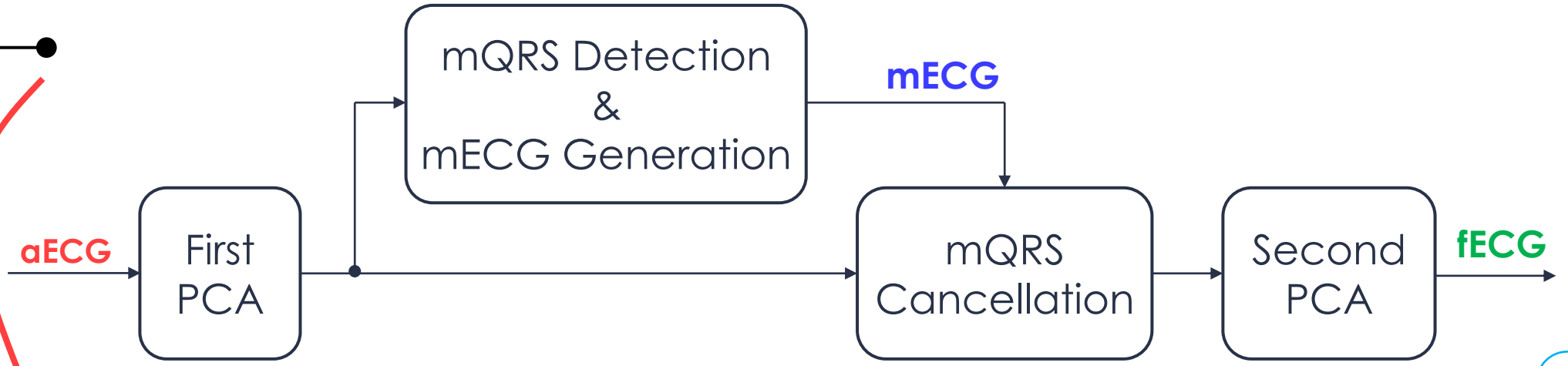


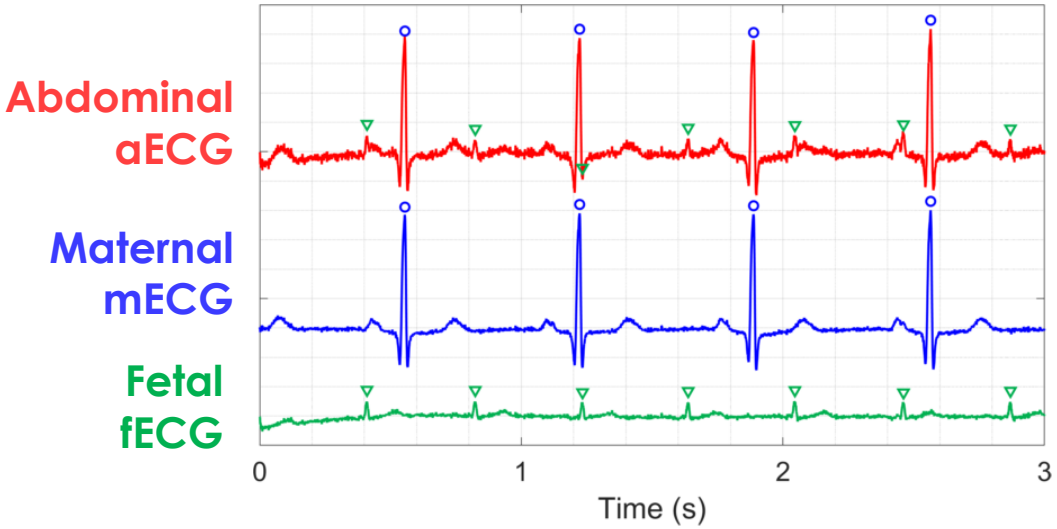


EPFL :: csem

Blind Source Separation for Fetal ECG

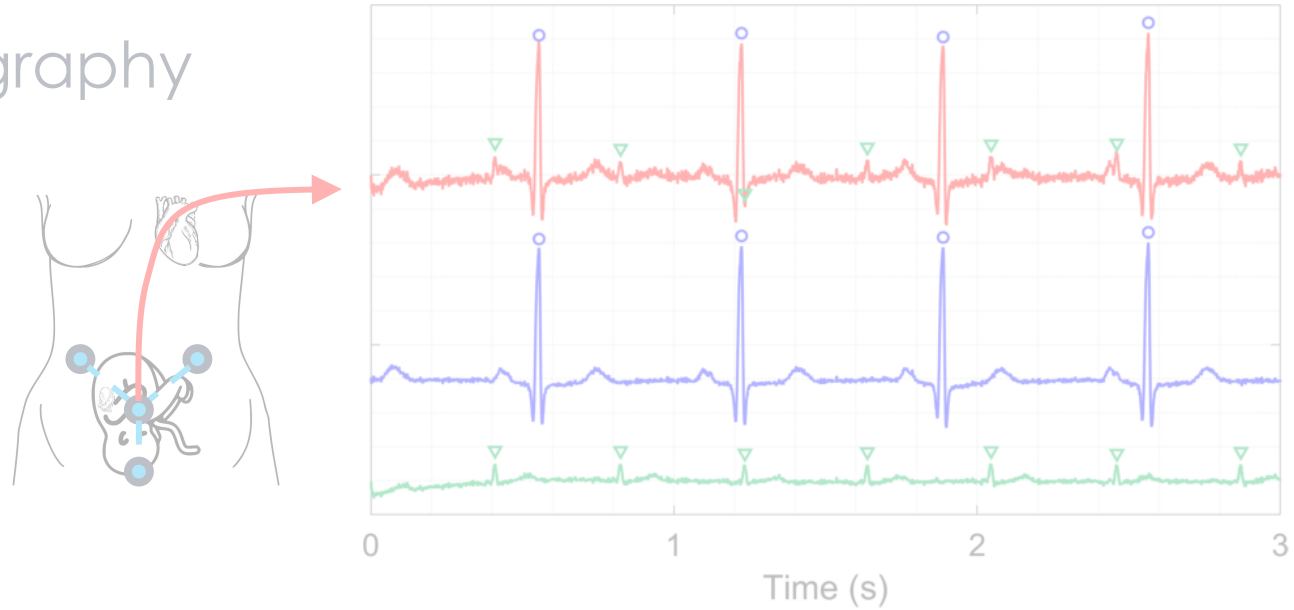




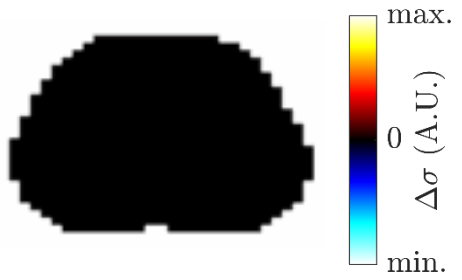


Lab #2

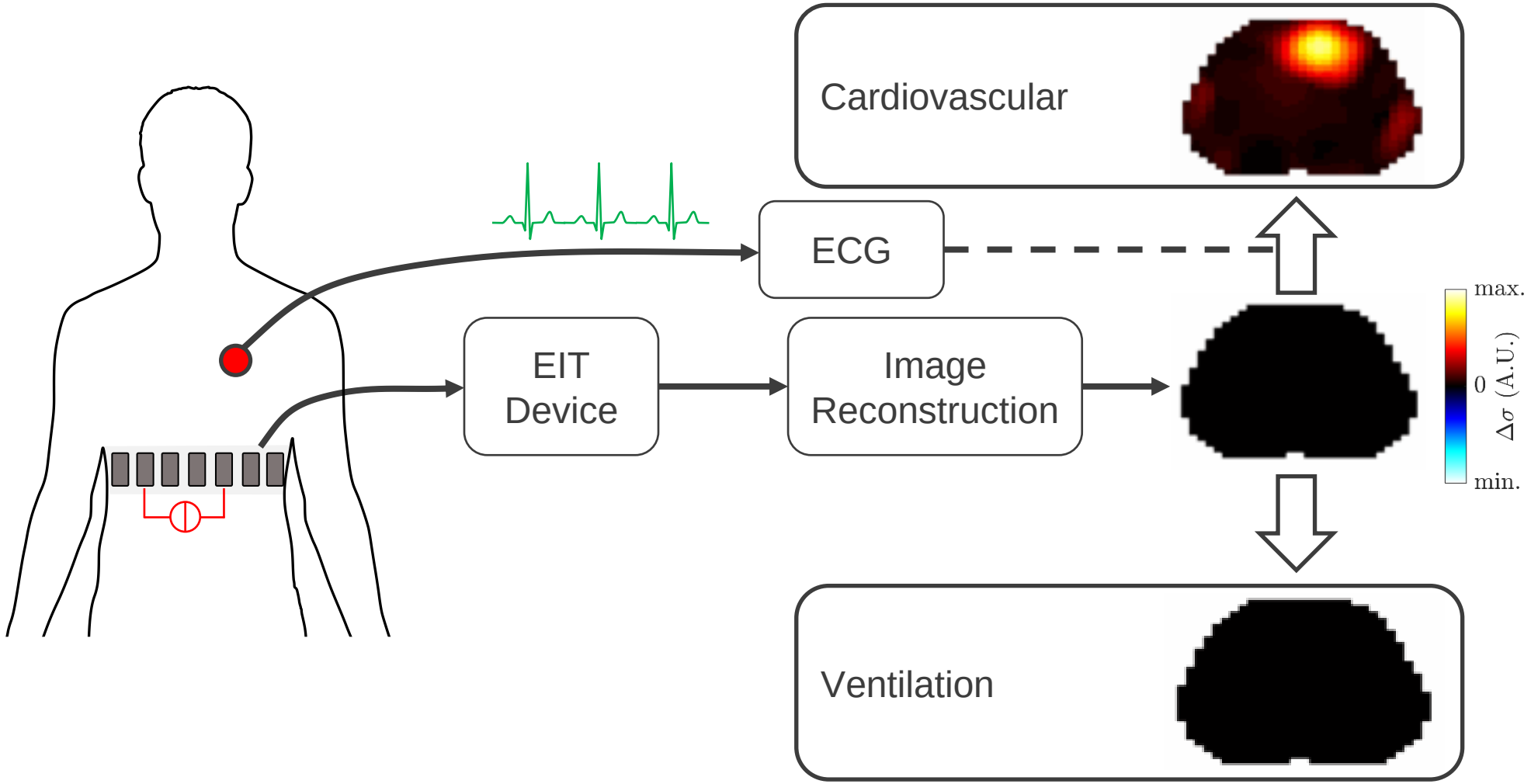
1. fECG: Fetal Electrocardiography



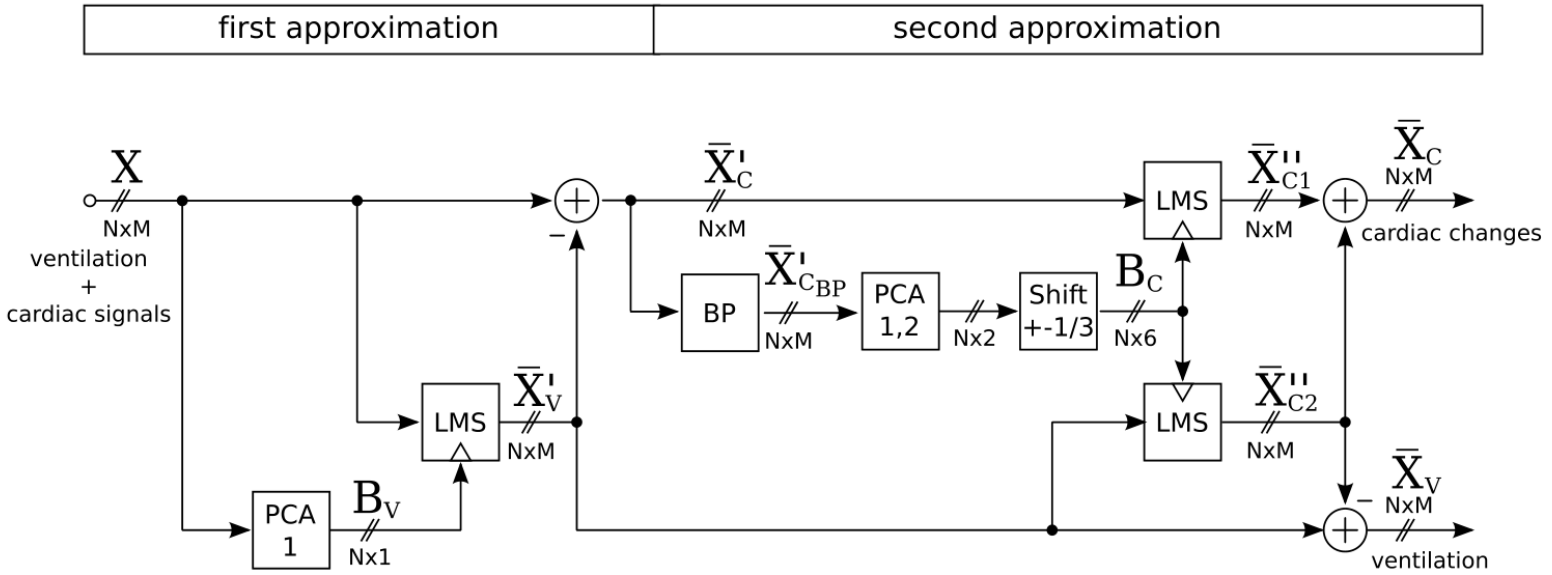
2. EIT: Electrical Impedance Tomography



EIT – Electrical Impedance Tomography



EIT – Electrical Impedance Tomography



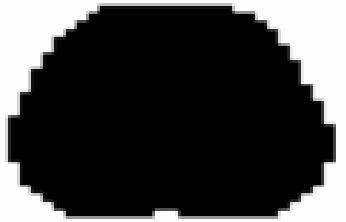
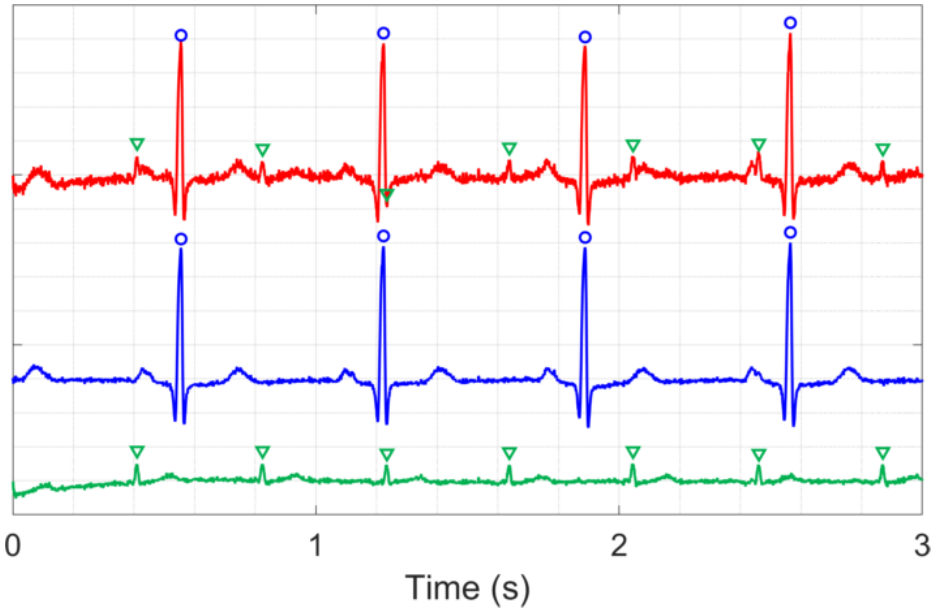


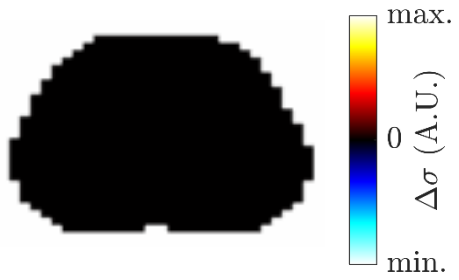
Figure 1. Flowchart of the method. PCA #: principal component number #, LMS: least-mean-squares fit, BP: bandpass, 'shift $\pm\frac{1}{3}$': input is phase shifted by $[-1/3,\ 0,\ 1/3]$ heart cycle to account for phase shifts introduced by the blood flow. *N*: number of processed frames, *M*: number of pixels in one frame.

J. M. Deibele, et al., "Dynamic separation of pulmonary and cardiac changes in electrical impedance tomography," *Physiological Measurement*, vol. 29, no. 6, pp. S1–S14, Jun. 2008, doi: 10.1088/0967-3334/29/6/S01.

Today's Lab – Instructions

- Please **submit** your **report** as a **single PDF file**.
- We recommend working in **groups of 3 students**; the last group can be a group of 2 or 4.
- You can **prepare one single report** for the group (*name1_name2_name3_lab_PCA.pdf*), but **every member needs to upload** the same file individually.
- There are **2 experiments** in this practical session. The **Python** code for each experiment is already coded and will be provided as **Jupyter notebooks**. These will only **require minimal input** from you. A major part of this practical session is thus focused on questions testing your understanding and correct interpretation of the signals and the analysis results that you see.





Karen Adam – kam@csem.ch

CSEM Signal Processing and AI Group