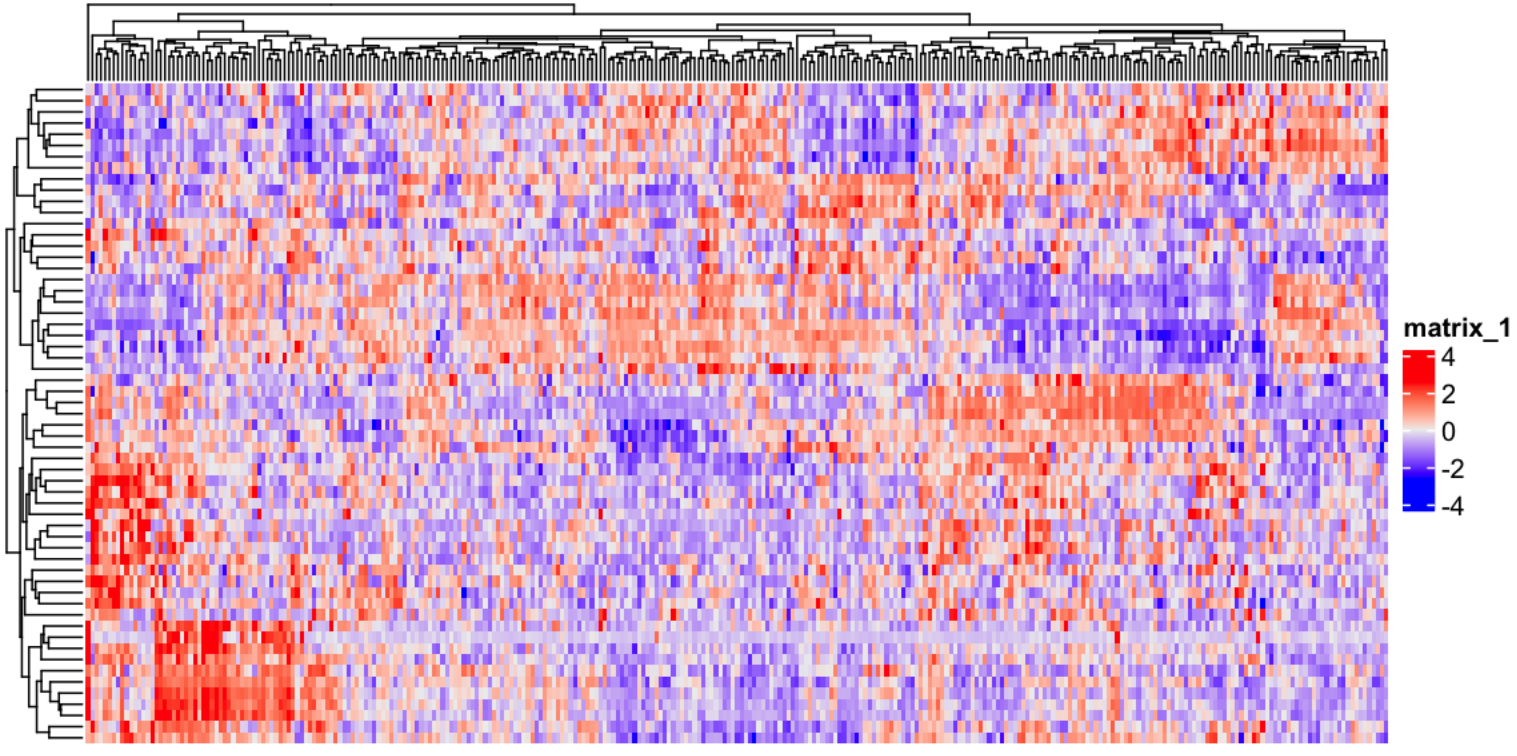
L8

Dimensionality Reduction & Data Visualization

Seungchan Kim
Center for Computational Systems Biology
Electrical and Computer Engineering

Heatmap



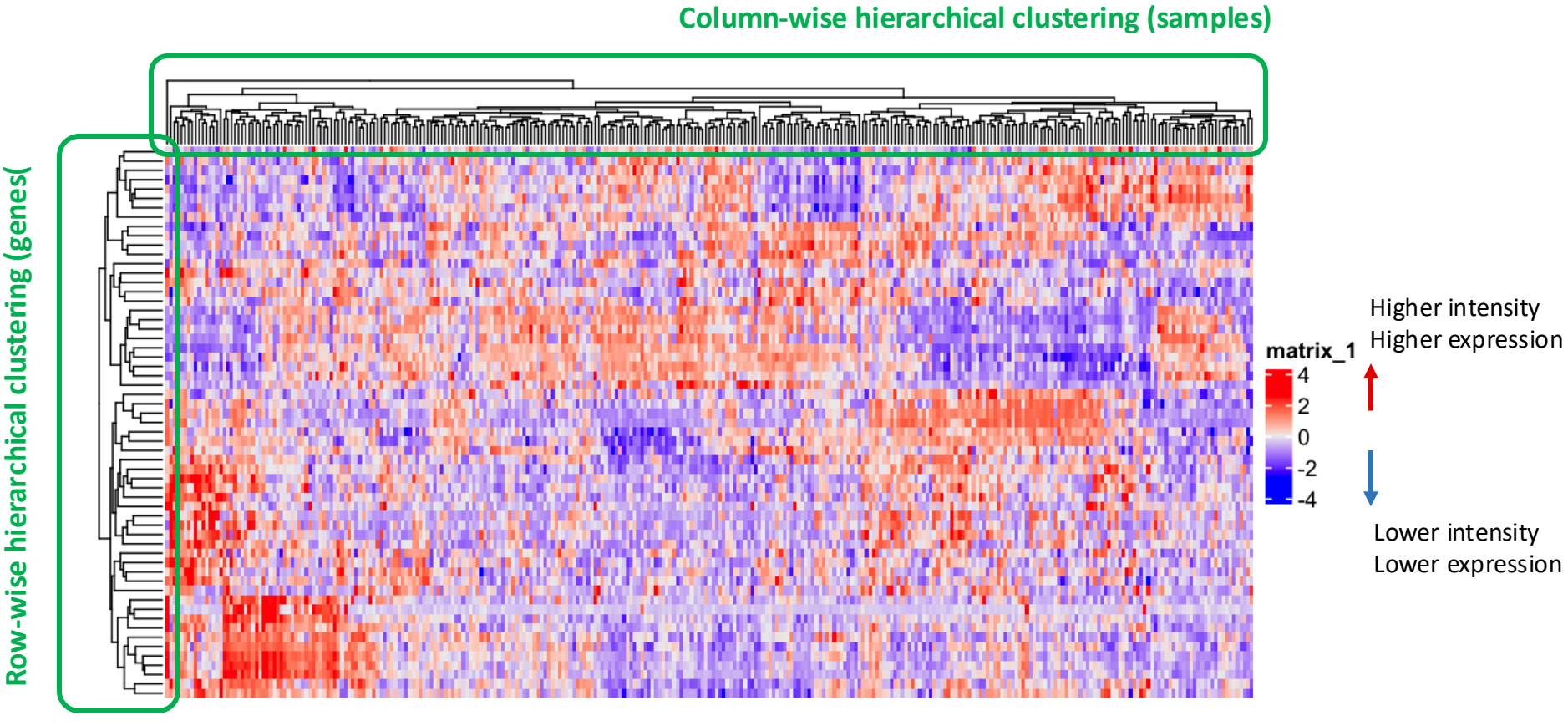
Heatmap

- Gene expression data are 2D matrix
  - Rows are genes/features and columns are observations/samples.
  - Note: this is the opposite of conventional data representation
- Gene expression data are often visualized in heatmap
  - Data are often normalized across rows (=genes) or columns (=samples) to highlight the difference across samples or genes.
- Available R Packages:
  - ComplexHeatmap
  - pheatmap

Heatmap

- Display the express of many genes across many samples
- Often used together with hierarchical clustering to reveal expression patterns
  - Row-wise clustering to group genes with similar expression patterns
  - Colum-wise clustering to group samples with similar expression patterns
- Color and intensity of tiles represent gene expression, often changes of gene expression

Heatmap


Column-wise hierarchical clustering (samples)
Row-wise hierarchical clustering (genes(
matrix_1
4
2
0
-2
-4
Higher intensity
Higher expression
Lower intensity
Lower expression

Challenges in RNAseq Data Visualization

- High dimensionality: >10,000 features (genes)

- Dimensionality reduction can help reduce computational complexity but still retain most of underlying structure in the data

Dimensionality Reduction

- What is the objective?
  - Choose an optimum set of features d* of lower dimensionality with most of underlying structure in the data retained
  - Data with reduced dimensionality can be used for visualization and discovery of underlying patterns/structure
- Different methods can be used to reduce dimensionality:
  - Feature extraction
  - Feature selection

Dimensionality Reduction (cont'd)

Feature extraction: computes a new set of features from the original features through some transformation f().

f() could be linear or non-linear

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ . \\ . \\ . \\ . \\ . \\ x_D \end{bmatrix} \xrightarrow{f(\mathbf{x})} \mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ . \\ . \\ y_K \end{bmatrix}$$

K<<D

Feature selection: chooses a subset of the original features.

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ . \\ . \\ . \\ . \\ . \\ x_D \end{bmatrix} \rightarrow \mathbf{y} = \begin{bmatrix} x_{i_1} \\ x_{i_2} \\ . \\ . \\ x_{i_K} \end{bmatrix}$$

K<<D

Feature Extraction

- Linear transformations are particularly attractive because they are simpler to compute and analytically tractable.
- Given $x \in R^D$, find an K x D matrix T such that:

$y = Tx \in R^K$ where K<<D

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ . \\ . \\ . \\ . \\ . \\ x_D \end{bmatrix} \xrightarrow{f(\mathbf{x})} \mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ . \\ . \\ y_K \end{bmatrix}$$

T

This is a projection transformation from D dimensions to K dimensions.

Each new feature y_i is a linear combination of the original features x_j

Feature Extraction (cont'd)

- From a mathematical point of view, finding an optimum mapping $y = f(x)$ can be formulated as an optimization problem (i.e., minimize or maximize an objective criterion).

- Commonly used objective criteria:

  - Minimize Information Loss: projection in the lower-dimensional space preserves as much information in the data as possible.

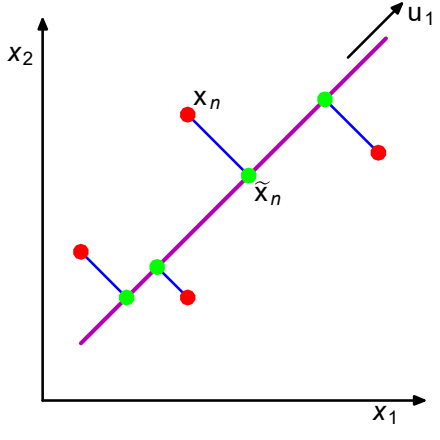
  - Maximize Discriminatory Information: projection in the lower-dimensional space increases class separability.

Feature Extraction (cont'd)

- Popular linear feature extraction methods:
  - Principal Components Analysis (PCA): Seeks a projection that minimizes information loss.
  - Linear Discriminant Analysis (LDA): Seeks a projection that maximizes discriminatory information.
- Many other methods:
  - Making features as independent as possible (Independent Component Analysis).
  - Retaining interesting directions (Projection Pursuit).
  - Embedding to lower dimensional manifolds (Isomap, Locally Linear Embedding).

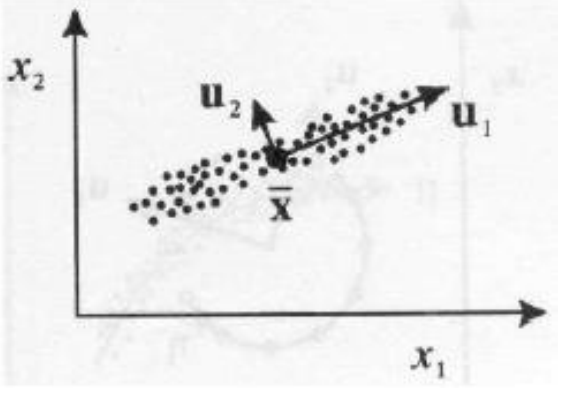
Principal Component Analysis

- PCA finds the linear subspace that
  - maximizes the explained variance
  - equivalently, minimizes the unexplained variance
- PCA can be applied to any multidimensional dataset
  - data do not have to be *Normally distributed*, a.k.a., *Gaussian*



Principal Component Analysis

- PCA chooses the eigenvectors corresponding to the largest eigenvalues.
- The eigenvalues correspond to the variance of the data along the eigenvector directions.
- Therefore, PCA projects the data along the directions where the data varies most.
- PCA preserves as much information in the data by preserving as much variance in the data.



u_1: direction of max variance
u_2: orthogonal to u_1

How should we choose K ?

- *K* is typically chosen based on how much information (variance) we want to preserve in the data:

Choose the smallest *K* that satisfies the following inequality:

$$\frac{\sum_i^K \lambda_i}{\sum_i^D \lambda_i} > T$$ where T is a threshold (e.g. 0.9)

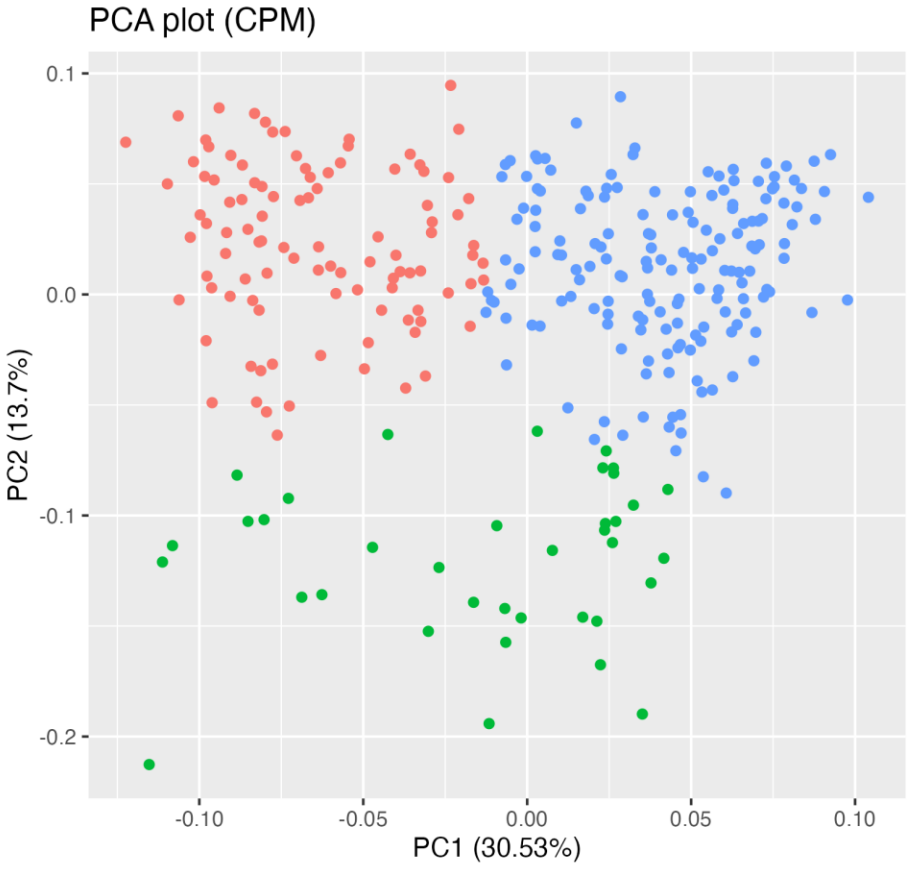
- If $T = 0.9$, for example, *K* is chosen to "preserve" 90% of the information (variance) in the data.
- If $K = D$, then we "preserve" 100% of the information in the data (i.e., just a "change" of basis and $\hat{\mathbf{x}} = \mathbf{x}$)

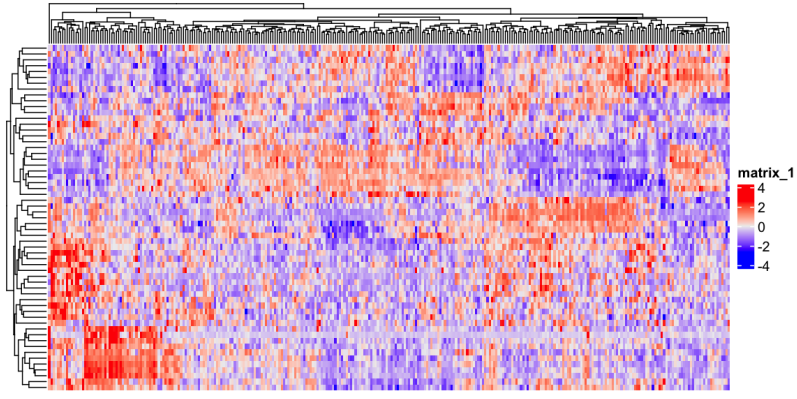
PCA plot for Gene Expression Data

- K (dimensionality) is typically set to 2 for visualization purpose

- However, for down-stream analysis, for example, to perform clustering analysis, it is set to retain the variability of the data as much as possible.

PCA: Example


PCA plot (CPM)
0.1
0.0
-0.1
-0.2
PC2 (13.7%)
-0.10
-0.05
0.00
0.05
0.10
PC1 (30.53%)


matrix_1
4
2
0
-2
-4


cluster
1
2
3

Other applications of PCA

- PCA reduces the dimensionality of the input space, hence, reducing the number of parameters that must be learned for classification or regression.
- This will help to reduce overlearning.

- However, there are other applications of PCA

Standardization

- Input vectors are often heterogeneous in that values might vary widely on some dimensions relative to others.
- This is particularly true when input vectors are composed of different kinds of measurements, perhaps measured in different units.
- Example:
  - We may try to classify a patient in a hospital setting based upon:
    - Age (years)
    - Resting pulse (beats per minute)
    - Body temperature (degrees Celsius)
    - …
- For pattern recognition algorithms to work well, it is often important that the data be standardized along these different dimensions.

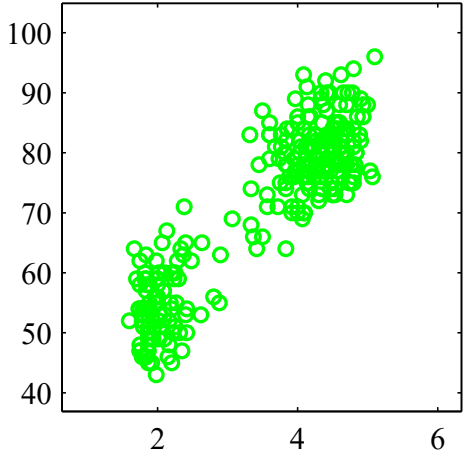
Pre-Whitening - decorrelation

- Let U be the $D \times D$ matrix whose columns are the D orthonormal eigenvectors $\mathbf{u}_i$ of $\mathbf{S}$.
- Let Λ be the $D \times D$ diagonal matrix whose diagonal elements Λ_{ii} are the associated eigenvalues λ_i.
- Then the transformation

$$\mathbf{y}_n = \Lambda^{-\frac{1}{2}} U^t (\mathbf{x}_\mathrm{n} - \bar{\mathbf{x}})$$

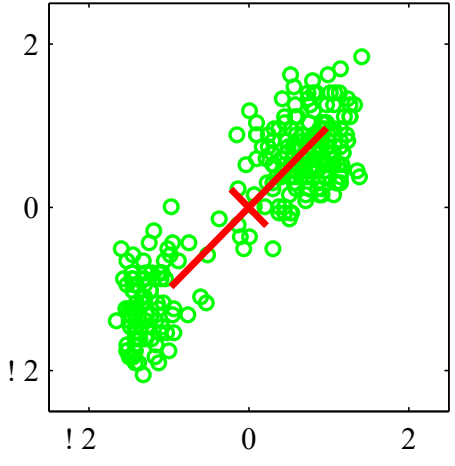
  does three things:
  - Shifts the data to the origin, so that the transformed data have zero mean.
  - Rotates the data into the principal axes, decorrelating the data (diagonal covariance)
  - Scales the data by the inverse standard deviation along each principal axis, thus normalizing the variance in all directions (covariance = identity matrix).

Pre-Whitening



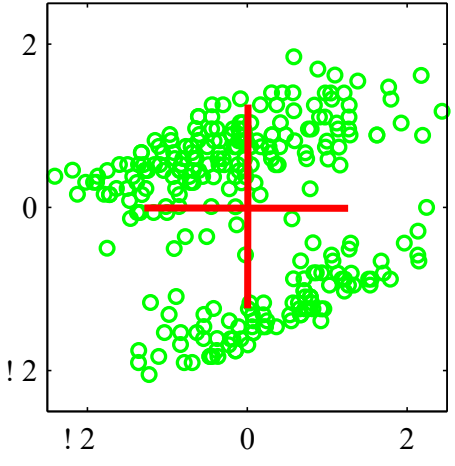
Original Data

$$y_i = \frac{x_i - \bar{x}_i}{\sigma_i}$$



Normalized to 0-mean
and unit variances (z-scores)

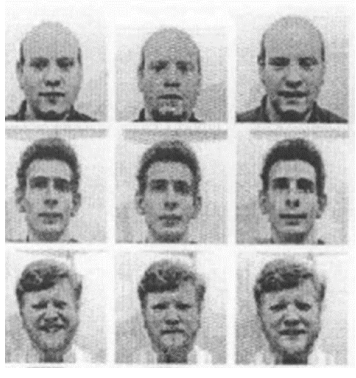
$$\mathbf{y}_n = \Lambda^{-1/2} U^t \left(\mathbf{x}_n - \bar{\mathbf{x}} \right)$$

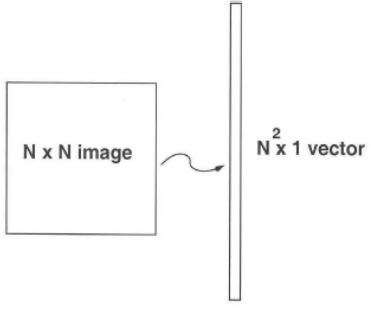


Whitened
(unit covariance)

Application to Images

- **Goal**: represent images in a space of lower dimensionality using PCA.
  - Useful for various applications, e.g., face recognition, image compression, etc.
- Given M images of size N x N, first represent each image as an N^2x1 1D vector (i.e., by stacking the rows together).





number of features:

$D=N^2$

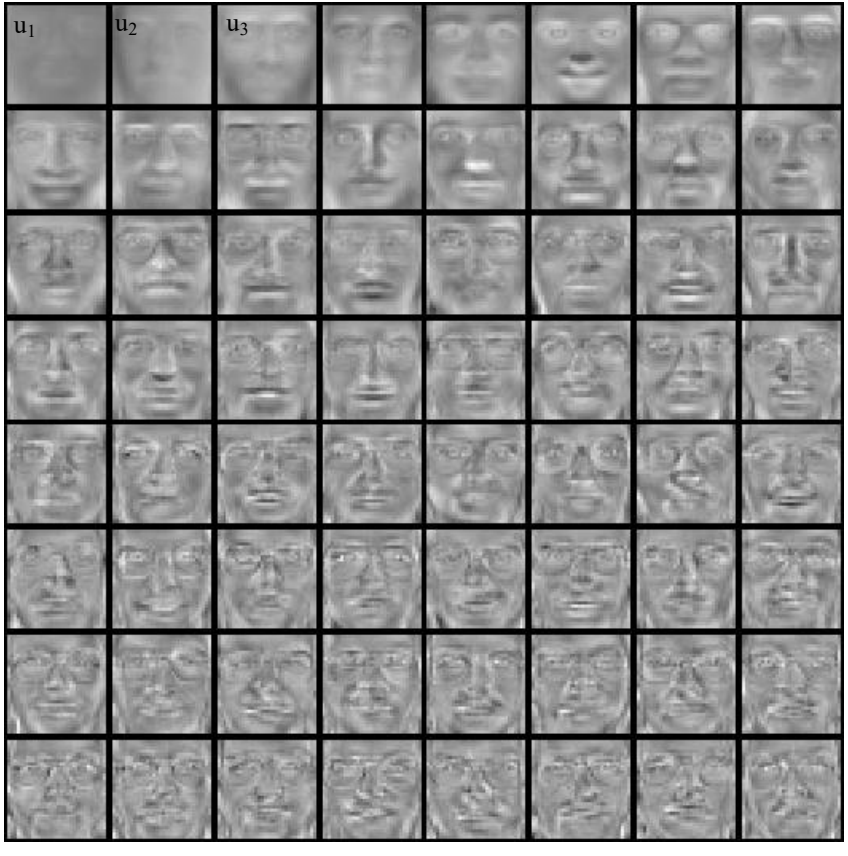
Example

Dataset

Example (cont'd)

Mean face: $\bar{\mathbf{x}}$

K largest eigenvectors: $u_1,\ldots u_K$
(visualized as images – called "eigenfaces")



Application to Images (cont'd)

- Interpretation: approximate a face image using eigenfaces

K largest eigenvectors: $u_1,\ldots u_K$ (basis vectors)

$$\hat{\mathbf{x}} = \sum_{i=1}^{K} y_i u_i = y_1 u_1 + y_2 u_2 + \ldots + y_K u_K + \bar{\mathbf{x}}$$

eigen-coefficients

$$\hat{\mathbf{x}} - \bar{\mathbf{x}}: \begin{bmatrix} y_1 \\ y_2 \\ . \\ . \\ y_K \end{bmatrix}$$