# Principal Component Analysis

Oliver Schulte - CMPT 880

# Outline

# Outline

# PCA: Motivation and Intuition

- Basic Ideas over 100 years old (stats). Still useful!
- Intuition: Suppose that your data is generated by a few hidden causes or factors. Then you could compactly describe each data point by how much each cause contributes to generate it.
- Principal Component Analysis (PCA) assumes that the contribution of each factor to each data point is *linear*.
- Data Columns are linear combinations of each other, can be merged.

# Informal Example: Student Performance

- Each student's performance is summarized in 4 assignments, 1 midterm, 1 project = 6 numbers.
- Suppose that on each item, a student's performance can be explained in terms of two components.
  - Her intelligence $I_n$
  - Her diligence $D_n$.
  - Combine these into a vector $z_n$.
- The importance of each component varies with the assignment. So we have 6 numbers for each. Put them in a 6x2 matrix $\boldsymbol{W}$.
- Then the performance numbers of student $n$ can be predicted by the model

$$\boldsymbol{x}_n = \boldsymbol{W}\boldsymbol{z}_n + \varepsilon,$$

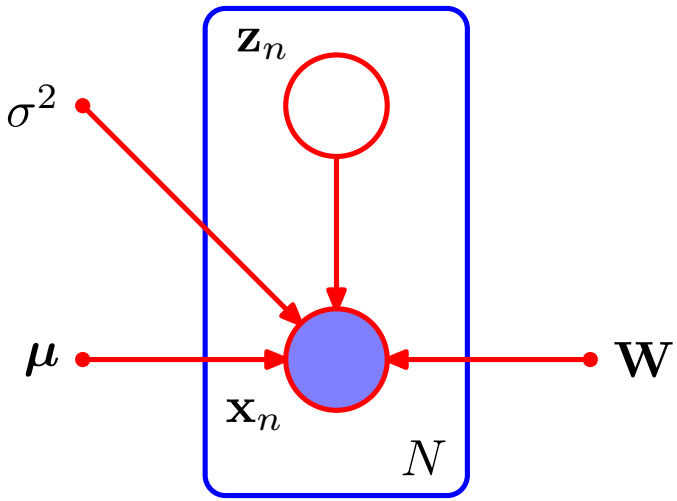
where $\varepsilon$ is (Gaussian) noise.

# Informal Example: Blind Source Separation

- Two people are talking in a room, sometimes at the same time. `http://www.youtube.com/watch?v=Qr74sM7oqQc&feature=related`
- Two microphones are set up at different parts of the room. Each mike catches each person from a different position. Let $x_i$ be the combined signal at microphone $i$.
- The contribution of person 1 to mike $i$ depends on the position of mike $i$, can be summarized as a pair of numbers $w_{i1}, w_{i2}$.
- Similarly for person 2. Combine these into a 2x2 matrix $\boldsymbol{W}$.
- Let $z_i$ be the (amplitude of) the voice signal of person $i$. Then the combined signal at mike 1 is given by $x_1 = w_{11} \cdot z_1 + w_{12} \cdot z_2$.
- Similarly for mike 2. Overall, we have that

$$\boldsymbol{x} = \boldsymbol{W}\boldsymbol{z}.$$

# Probabilistic PCA

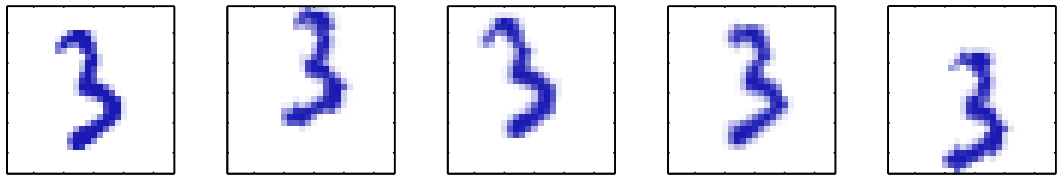



- Probabilistic model of PCA: For each data point $\boldsymbol{x}_n$, there is a latent variable vector $\boldsymbol{z}_n$.
- Linear Gaussian model:

$$\boldsymbol{x} = \boldsymbol{W}\boldsymbol{z} + \boldsymbol{\mu} + \boldsymbol{\varepsilon}.$$

- Can train using EM.
- Handles missing data.
- Can take mixtures of PCA models.
- Closely related to **factor analysis.** `http://cscs.umich.edu/~crshalizi/weblog/`.

# Example: Digit Rotation



- Take a single digit (3), make 100x100 pixel image
- Create multiple copies by translating (vertical/horizontal) and rotating
- This dataset could be represented as vectors in $\mathbb{R}^{100x100} = \mathbb{R}^{10000}$
- But the dataset only has 3 degrees of freedom... why are 10,000 needed?
  - Shouldn't a manifold or subspace of intrinsic dimension 3 suffice?

# Component Analysis: Pros and Cons

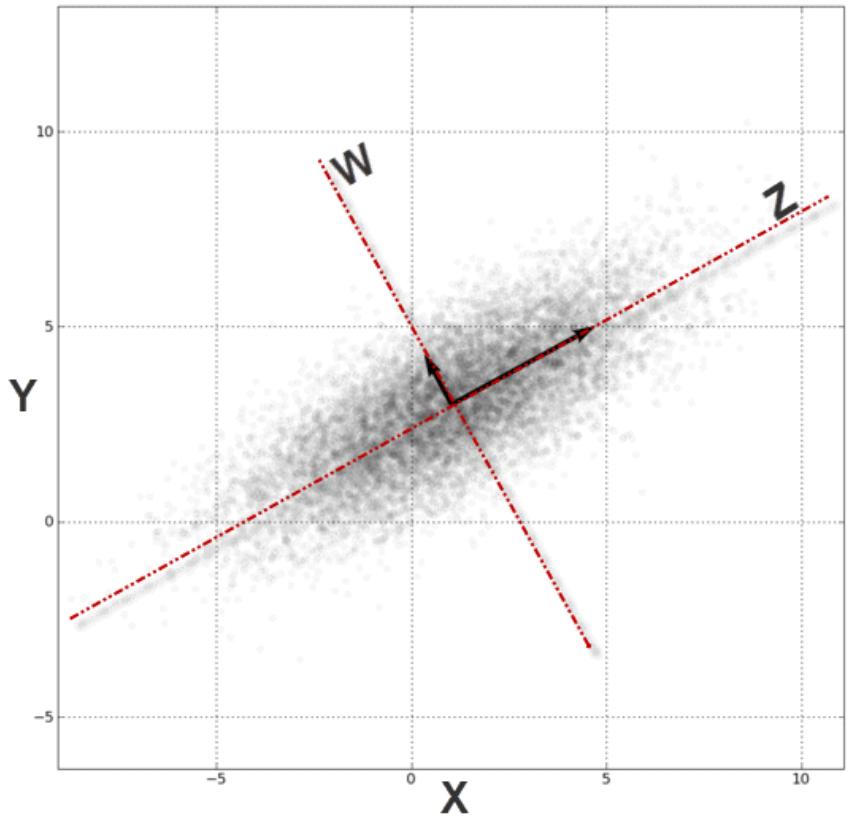
Pros

- Reduces dimensionality of data: easier to learn.
- Removes noise, filters out important regularities.
- Can be used to standardize data (whitening).

Cons

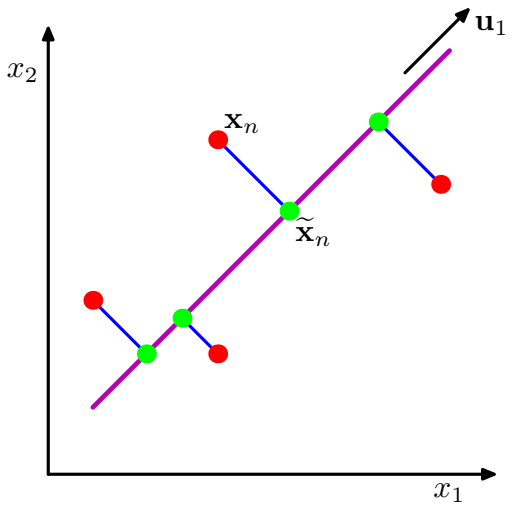
- PCA is restricted to linear hidden models. (Relax later).
- Black box: data vectors become hard to interpret.

# Pre-processing Example



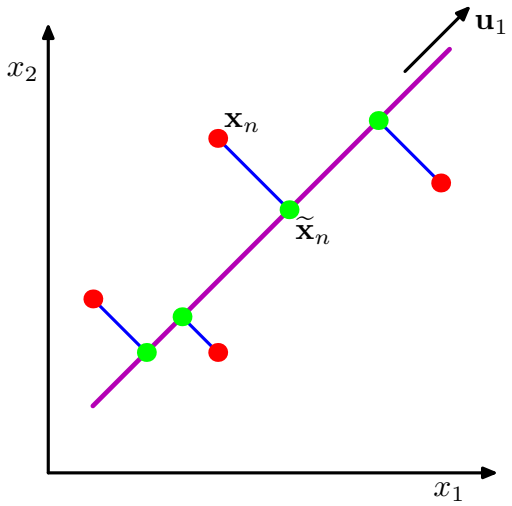

- In the Z-W coordinate sytem, we obtain a data set with mean 0 and unit covariance matrix.
- Correlations between dimensions have been removed.

# Dimensionality Reduction




- PCA finds a lower dimensional linear space to represent data
- How to define the right linear space?
  - Subspace that maximizes variance of projected data
  - Minimizes projection cost
- Turns out they are the same!
- Teapot demo `http://www.youtube.com/watch?v=BfTMmoDFXyE`

# Maximum Variance




- Consider dataset $\{\boldsymbol{x}_n \in \mathbb{R}^D\}$
- Try to project into space with dimensionality $M < D$
- For $M = 1$, space given by $\boldsymbol{u}_1 \in \mathbb{R}^D$, $\boldsymbol{u}_1^T \boldsymbol{u}_1 = 1$
- Optimization problem: find $\boldsymbol{u}_1$ that maximizes variance

# Projected variance

- The projection of a datapoint $\boldsymbol{x}_n \in \mathbb{R}^D$ by $\boldsymbol{u}_1$ is $\boldsymbol{u}_1^T \boldsymbol{x}_n$
- The mean of the projected data is

$$\frac{1}{N} \sum_{n=1}^{N} \boldsymbol{u}_1^T \boldsymbol{x}_n = \boldsymbol{u}_1^T \left( \frac{1}{N} \sum_{n=1}^{N} \boldsymbol{x}_n \right) = \boldsymbol{u}_1^T \bar{\boldsymbol{x}}$$

- The variance of the projected data is

$$\begin{aligned} \frac{1}{N} \sum_{n=1}^{N} \left\{ \boldsymbol{u}_1^T \boldsymbol{x}_n - \boldsymbol{u}_1^T \bar{\boldsymbol{x}} \right\}^2 &= \boldsymbol{u}_1^T \left( \frac{1}{N} \sum_{n=1}^{N} (\boldsymbol{x}_n - \bar{\boldsymbol{x}})(\boldsymbol{x}_n - \bar{\boldsymbol{x}})^T \right) \boldsymbol{u}_1 \\ &= \boldsymbol{u}_1^T \boldsymbol{S} \boldsymbol{u}_1 \end{aligned}$$

where $\boldsymbol{S}$ is the sample covariance.

# Optimization

- How do we maximize the projected variance $\boldsymbol{u}_1^T \boldsymbol{S} \boldsymbol{u}_1$ subject to the constraint that $\boldsymbol{u}_1^T \boldsymbol{u}_1 = 1$?
  - Lagrange multipliers:

$$\boldsymbol{u}_1^T \boldsymbol{S} \boldsymbol{u}_1 + \lambda_1 (1 - \boldsymbol{u}_1^T \boldsymbol{u}_1)$$

  - Taking derivatives, stationary point when

$$\boldsymbol{S} \boldsymbol{u}_1 = \lambda_1 \boldsymbol{u}_1$$

    i.e. $\boldsymbol{u}_1$ is an eigenvector of $\boldsymbol{S}$

# Optimization – Which Eigenvector

- There are up to $D$ eigenvectors, which is the right one?
- Maximize variance!
- Variance is:

$$\begin{aligned} & \boldsymbol{u}_1^T \boldsymbol{S} \boldsymbol{u}_1 \\ = \quad & \boldsymbol{u}_1^T \lambda_1 \boldsymbol{u}_1 \text{ since } \boldsymbol{u}_1 \text{ is an eigenvector} \\ = \quad & \lambda_1 \text{ since } ||\boldsymbol{u}_1|| = 1 \end{aligned}$$

- Choose the eigenvector $\boldsymbol{u}_1$ corresponding to the largest eigenvalue $\lambda_1$
  - This is the first direction ($M = 1$)
  - If $M > 1$, simple to show eigenvectors corresponding to largest $M$ eigenvalues are the ones to choose to maximize projected variance

# Reconstruction Error

- Can also phrase problem as finding set of orthonormal basis vectors $\{\boldsymbol{u}_i\}$ for projection
- Find set of $M < D$ vectors to minimize reconstruction error

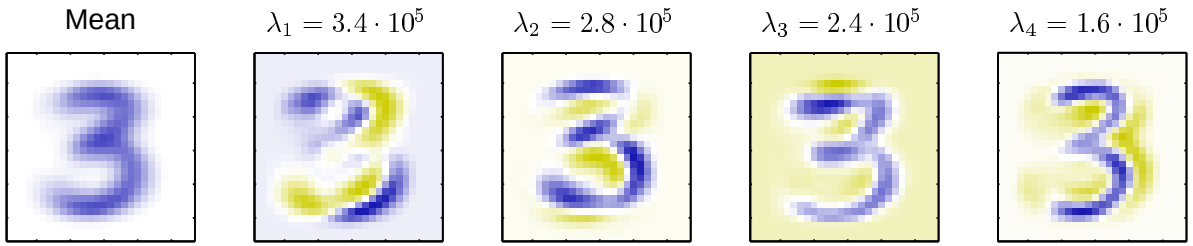
$$J = \frac{1}{N} \sum_{n=1}^{N} ||\boldsymbol{x}_n - \tilde{\boldsymbol{x}}_n||^2$$

where $\tilde{\boldsymbol{x}}_n$ is projected version of $\boldsymbol{x}_n$

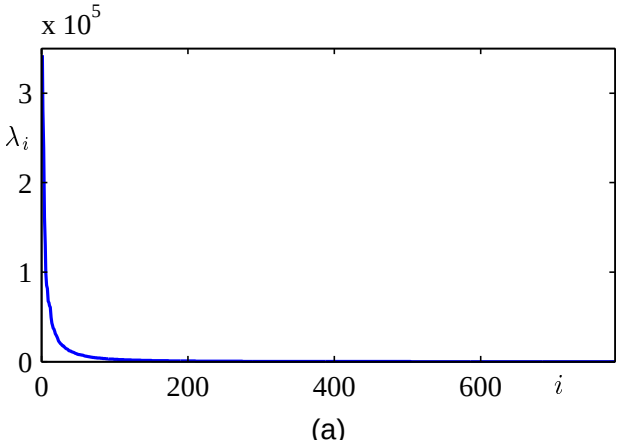
- $\tilde{\boldsymbol{x}}_n$ will end up being same as before – mean plus leading eigenvectors of covariance matrix $\boldsymbol{S}$

$$\tilde{\boldsymbol{x}}_n = \bar{\boldsymbol{x}} + \sum_{i=1}^{M} \beta_{ni} \boldsymbol{u}_i$$

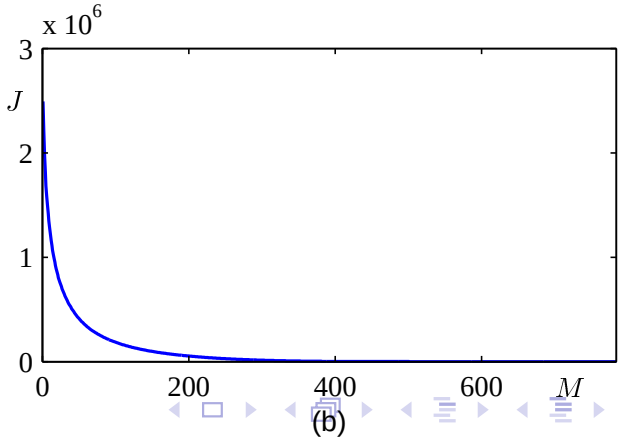
# PCA Example – MNIST Digits




- PCA of digits “3” from MNIST
- Each eigenvector has the same dimension as a data point.
- First $\approx 100$ dimensions capture most variance / low reconstruction error $J$

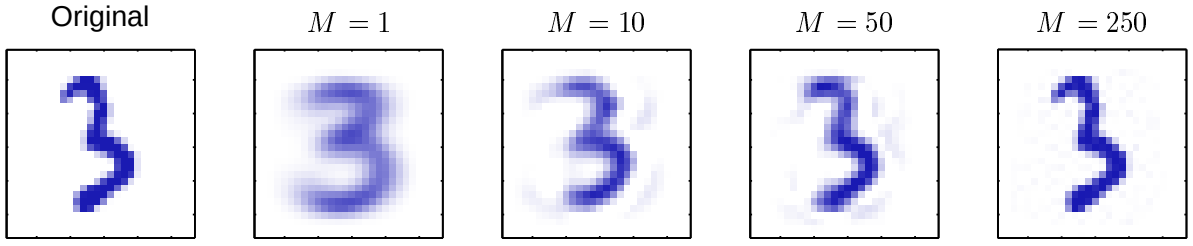


(a)



(b)

# Reconstruction – MNIST Digits

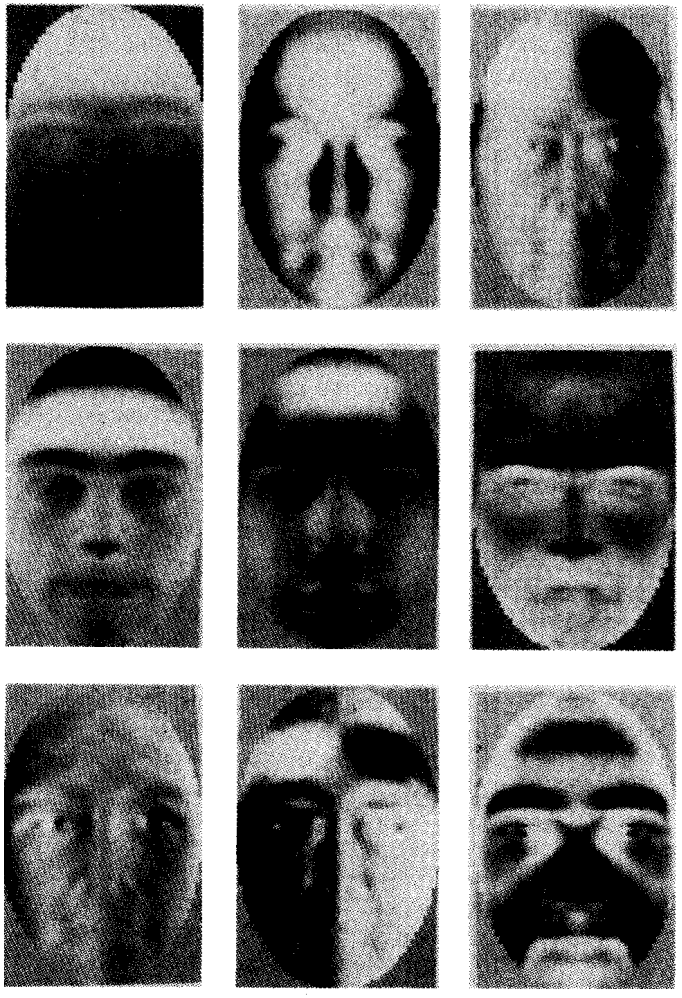



- PCA approximation to a data vector $\boldsymbol{x}_n$ is:

$$\tilde{\boldsymbol{x}}_n = \bar{\boldsymbol{x}} + \sum_{i=1}^{M} \beta_{ni} \boldsymbol{u}_i$$

- As $M$ is increased, this reconstruction becomes more accurate
- $D = 784$, but with $M = 250$ quite good reconstruction
  - Dimensionality reduction

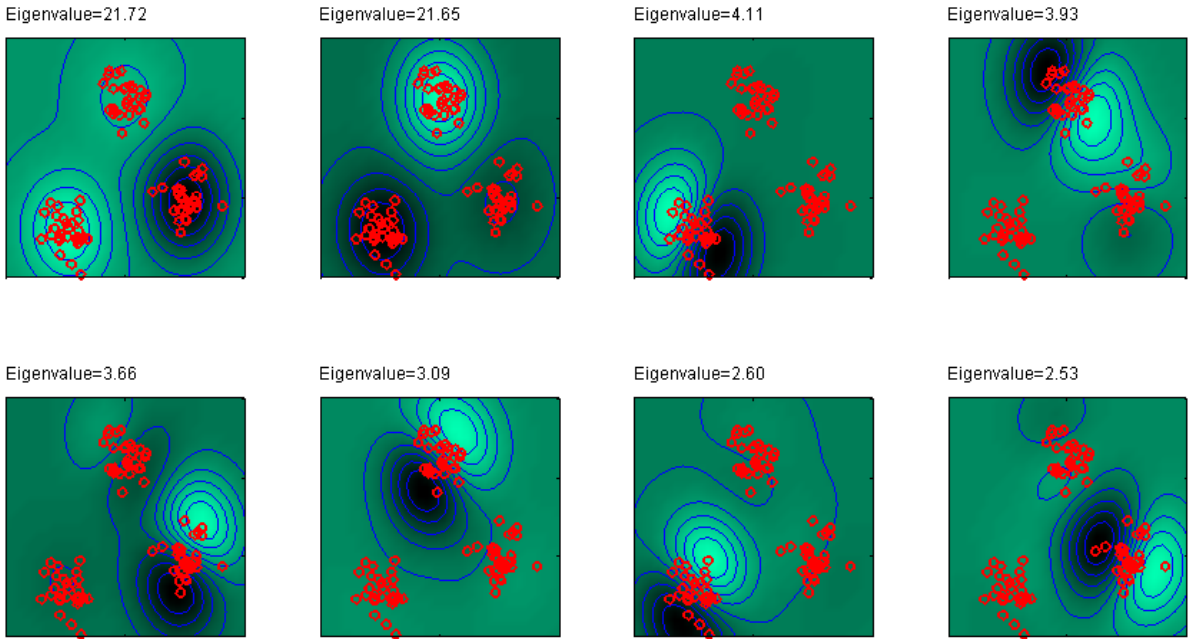
# PCA Example – Eigenfaces



Kirby and Sirovich, PAMI 1990

`http://en.wikipedia.org/wiki/Eigenface`.

# Nonlinear PCA: Kernel Methods




- Can use the kernel trick: replace dot products with kernel evaluations.
- Countour = constant projection on principeal component.
- In the figure, the first 2 eigenvectors separate 3 clusters.
- The next 3 split the clusters in halves.
- The last 3 split the clusters in orthogonal halves.

# Conclusion

- We discussed one method for finding a lower dimensional manifold – principal component analysis (PCA)
- PCA is a basic technique
  - Finds linear manifold (hyperplane)
- In general, manifold will be non-linear
  - Simple example – translating digit 1
  - Also “dimensions” corresponding to style of digit
- Other important techniques for dimensionality reduction: independent component analysis (ICA), isomap, locally linear embeddings