PCA / SVD

COMS 4721 Spring 2022

Daniel Hsu

PCA

Empirical (sample) covariance

Recall: If $\vec{X}$ is a random vector in $\mathbb{R}^d$, its covariance matrix is

$$\mathrm{cov}(\vec{X}) = \mathbb{E}[(\vec{X} - \mathbb{E}(\vec{X}))(\vec{X} - \mathbb{E}(\vec{X}))^{\mathsf{T}}].$$

Empirical (sample) covariance

Recall: If $\vec{X}$ is a random vector in $\mathbb{R}^d$, its covariance matrix is

$$\mathrm{cov}(\vec{X}) = \mathbb{E}[(\vec{X} - \mathbb{E}(\vec{X}))(\vec{X} - \mathbb{E}(\vec{X}))^{\mathsf{T}}].$$

If $\vec{x}_1, \ldots \vec{x}_n \in \mathbb{R}^d$ are n data points from $\mathbb{R}^d$, and P_n is the empirical distribution on these points, then their **empirical covariance matrix** (a.k.a. **sample covariance matrix**) is $\mathrm{cov}(\vec{X})$ for $\vec{X} \sim P_n$

More properties of the covariance matrix

Properties of $\mathrm{cov}(\vec{X})$:

More properties of the covariance matrix

Properties of $\mathrm{cov}(\vec{X})$:

- Symmetric

More properties of the covariance matrix

Properties of $\mathrm{cov}(\vec{X})$:

- Symmetric
- d real eigenvalues $\lambda_1, \ldots, \lambda_d$ (possibly repeated)
  Convention: Order from largest to smallest

$$\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_d$$

More properties of the covariance matrix

Properties of $\mathrm{cov}(\vec{X})$:

- Symmetric
- d real eigenvalues $\lambda_1, \ldots, \lambda_d$ (possibly repeated)
  Convention: Order from largest to smallest

$$\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_d$$

- Can choose corresponding eigenvectors $\vec{v}_1, \vec{v}_2, \ldots, \vec{v}_d$ s.t. they are orthonormal

More properties of the covariance matrix

Properties of $\operatorname{cov}(\vec{X})$:

- Symmetric
- d real eigenvalues $\lambda_1, \ldots, \lambda_d$ (possibly repeated)
  Convention: Order from largest to smallest

$$\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_d$$

- Can choose corresponding eigenvectors $\vec{v}_1, \vec{v}_2, \ldots, \vec{v}_d$ s.t. they are orthonormal
- In fact, all eigenvalues are non-negative:

$$\vec{v}_i^{\mathsf{T}} \operatorname{cov}(\vec{X}) \vec{v}_i = \vec{v}_i^{\mathsf{T}} (\lambda_i \vec{v}_i) = \lambda_i$$

and

$$\vec{v}_i^{\mathsf{T}} \operatorname{cov}(\vec{X}) \vec{v}_i = \operatorname{var}(\vec{v}_i \cdot \vec{X}) \geq 0$$

More properties of the covariance matrix

Properties of $\operatorname{cov}(\vec{X})$:

- Symmetric
- d real eigenvalues $\lambda_1, \ldots, \lambda_d$ (possibly repeated)
  Convention: Order from largest to smallest

$$\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_d$$

- Can choose corresponding eigenvectors $\vec{v}_1, \vec{v}_2, \ldots, \vec{v}_d$ s.t. they are orthonormal
- In fact, all eigenvalues are non-negative:

$$\vec{v}_i^{\mathsf{T}} \operatorname{cov}(\vec{X}) \vec{v}_i = \vec{v}_i^{\mathsf{T}} (\lambda_i \vec{v}_i) = \lambda_i$$

and $$\vec{v}_i^{\mathsf{T}} \operatorname{cov}(\vec{X}) \vec{v}_i = \operatorname{var}(\vec{v}_i \cdot \vec{X}) \geq 0$$

Symmetric matrices with non-negative eigenvalues are said to be **positive semidefinite (PSD)**

More properties of the covariance matrix

Properties of $\mathrm{cov}(\vec{X})$:

- Symmetric
- d real eigenvalues $\lambda_1, \ldots, \lambda_d$ (possibly repeated)
  Convention: Order from largest to smallest

$$\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_d$$

- Can choose corresponding eigenvectors $\vec{v}_1, \vec{v}_2, \ldots, \vec{v}_d$ s.t. they are orthonormal
- In fact, all eigenvalues are non-negative:

$$\vec{v}_i^{\mathsf{T}} \mathrm{cov}(\vec{X}) \vec{v}_i = \vec{v}_i^{\mathsf{T}} (\lambda_i \vec{v}_i) = \lambda_i$$

and $$\vec{v}_i^{\mathsf{T}} \mathrm{cov}(\vec{X}) \vec{v}_i = \mathrm{var}(\vec{v}_i \cdot \vec{X}) \geq 0$$

  Symmetric matrices with non-negative eigenvalues are said to be **positive semidefinite (PSD)**

- **Eigenvalue decomposition** (a.k.a. **spectral decomposition**) of $\mathrm{cov}(\vec{X})$:

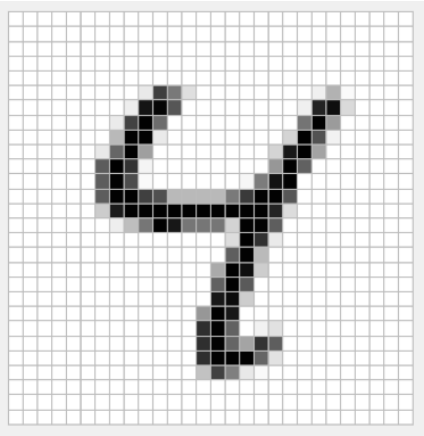
$$\mathrm{cov}(\vec{X}) = \sum_{i=1}^{d} \lambda_i \vec{v}_i \vec{v}_i^{\mathsf{T}}$$

Example: MNIST data set of handwritten digits

60000 images of handwritten digits, each a 28×28 gray-scale pixel image

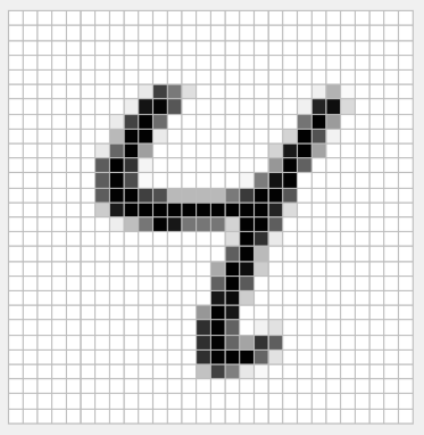
Example: MNIST data set of handwritten digits

Each image is represented as vector in $[0, 255]^d$ for $d = 28 \times 28 = 784$, one feature per pixel



Example: MNIST data set of handwritten digits

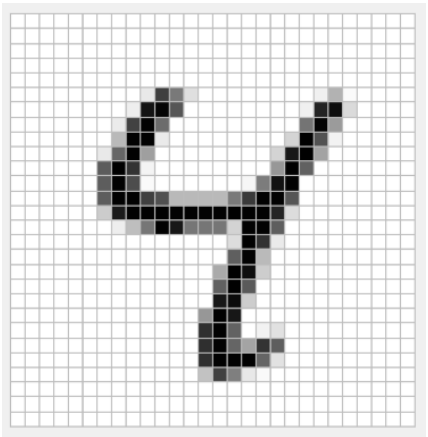
Each image is represented as vector in $[0, 255]^d$ for $d = 28 \times 28 = 784$, one feature per pixel



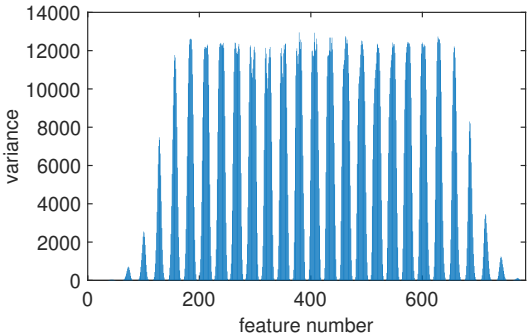
Some "features" (pixels) are more variable than others (on average over the data set)

Example: MNIST data set of handwritten digits

Each image is represented as vector in $[0, 255]^d$ for $d = 28 \times 28 = 784$, one feature per pixel

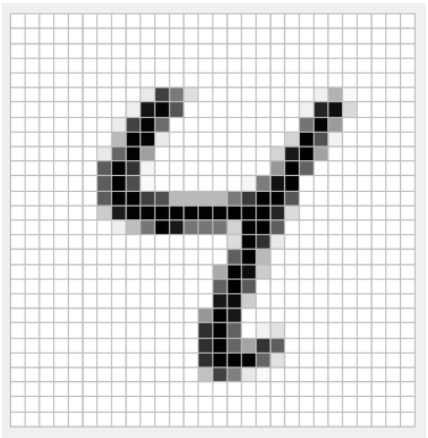


Some "features" (pixels) are more variable than others (on average over the data set)

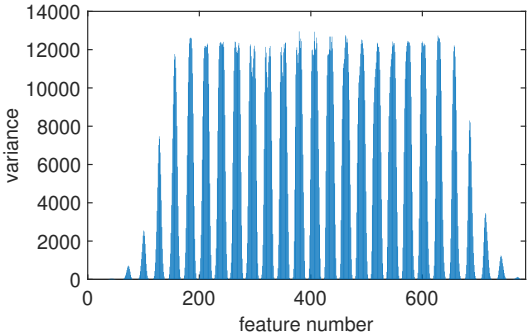


Example: MNIST data set of handwritten digits

Each image is represented as vector in $[0, 255]^d$ for $d = 28 \times 28 = 784$, one feature per pixel



Some "features" (pixels) are more variable than others (on average over the data set)



Perhaps not all features are necessary!

Data approximation

Coarse approximation: (our baseline)
Represent each data point by the overall mean $\vec{\mu}$
(where $\vec{\mu} := \mathbb{E}(\vec{X})$)

$$\vec{X} \longmapsto \vec{\mu}$$

Zero features per data point!

Data approximation

Coarse approximation: (our baseline)
Represent each data point by the overall mean $\vec{\mu}$
(where $\vec{\mu} := \mathbb{E}(\vec{X})$)

$$\vec{X} \longmapsto \vec{\mu}$$

Zero features per data point!

Expected reconstruction error:

$$\mathbb{E}\left[\|\vec{X} - \vec{\mu}\|_2^2\right]$$

Data approximation

Coarse approximation: (our baseline)
Represent each data point by the overall mean $\vec{\mu}$ (where $\vec{\mu} := \mathbb{E}(\vec{X})$)

$$\vec{X} \longmapsto \vec{\mu}$$

Zero features per data point!

Expected reconstruction error:

$$\mathbb{E}\left[\|\vec{X} - \vec{\mu}\|_2^2\right]$$

Less coarse approximation:
Represent each data point by overall mean $\vec{\mu}$ plus projection of $\vec{X} - \vec{\mu}$ to subspace $W \subseteq \mathbb{R}^d$

$$\vec{X} \longmapsto \vec{\mu} + \Pi_W(\vec{X} - \vec{\mu})$$

$k := \dim(W)$ features per data point

Data approximation

Coarse approximation: (our baseline)
Represent each data point by the overall mean $\vec{\mu}$ (where $\vec{\mu} := \mathbb{E}(\vec{X})$)

$$\vec{X} \longmapsto \vec{\mu}$$

Zero features per data point!

Expected reconstruction error:

$$\mathbb{E}\left[\|\vec{X} - \vec{\mu}\|_2^2\right]$$

Less coarse approximation:
Represent each data point by overall mean $\vec{\mu}$ plus projection of $\vec{X} - \vec{\mu}$ to subspace $W \subseteq \mathbb{R}^d$

$$\vec{X} \longmapsto \vec{\mu} + \Pi_W(\vec{X} - \vec{\mu})$$

$k := \dim(W)$ features per data point

Expected reconstruction error:

$$\mathbb{E}\left[\|\vec{X} - (\vec{\mu} + \Pi_W(\vec{X} - \vec{\mu}))\|_2^2\right]$$

Data approximation

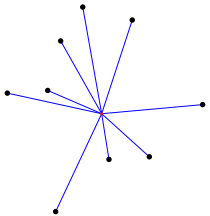
Coarse approximation: (our baseline)
Represent each data point by the overall mean $\vec{\mu}$ (where $\vec{\mu} := \mathbb{E}(\vec{X})$)

$$\vec{X} \longmapsto \vec{\mu}$$

Zero features per data point!

Expected reconstruction error:

$$\mathbb{E}\left[\|\vec{X} - \vec{\mu}\|_2^2\right]$$



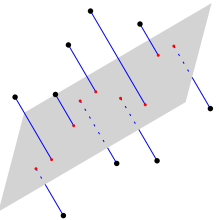
Less coarse approximation:
Represent each data point by overall mean $\vec{\mu}$ plus projection of $\vec{X} - \vec{\mu}$ to subspace $W \subseteq \mathbb{R}^d$

$$\vec{X} \longmapsto \vec{\mu} + \Pi_W(\vec{X} - \vec{\mu})$$

$k := \dim(W)$ features per data point

Expected reconstruction error:

$$\mathbb{E}\left[\|\vec{X} - (\vec{\mu} + \Pi_W(\vec{X} - \vec{\mu}))\|_2^2\right]$$



Special case: Coordinate subspaces

Suppose we only allow W to be a **coordinate subspace**, i.e., span of some coordinate vectors

Special case: Coordinate subspaces

Suppose we only allow W to be a **coordinate subspace**, i.e., span of some coordinate vectors

- Effect of $\vec{X} \longmapsto \vec{\mu} + \Pi_W(\vec{X} - \vec{\mu})$:

  Replace all but $k = \dim(W)$ of the features in $\vec{X}$ with corresponding mean value in $\vec{\mu}$

Special case: Coordinate subspaces

Suppose we only allow W to be a **coordinate subspace**, i.e., span of some coordinate vectors

- Effect of $\vec{X} \longmapsto \vec{\mu} + \Pi_W(\vec{X} - \vec{\mu})$:

  Replace all but $k = \dim(W)$ of the features in $\vec{X}$ with corresponding mean value in $\vec{\mu}$

- MNIST data: each "coordinate vector" corresponds to a pixel location $(i, j) \in [28] \times [28]$
  - For expected reconstruction error to be

$$10\% \times \underbrace{\mathbb{E}\left[\|\vec{X} - \vec{\mu}\|_2^2\right]}_{\text{our baseline}},$$

    need $k = 299$ features (pixels)

Special case: Coordinate subspaces

Suppose we only allow W to be a **coordinate subspace**, i.e., span of some coordinate vectors

- Effect of $\vec{X} \longmapsto \vec{\mu} + \Pi_W(\vec{X} - \vec{\mu})$:

  Replace all but $k = \dim(W)$ of the features in $\vec{X}$ with corresponding mean value in $\vec{\mu}$

- MNIST data: each "coordinate vector" corresponds to a pixel location $(i, j) \in [28] \times [28]$
  - For expected reconstruction error to be

$$10\% \times \underbrace{\mathbb{E}\left[\|\vec{X} - \vec{\mu}\|_2^2\right]}_{\text{our baseline}},$$

    need $k = 299$ features (pixels)

Question: Can we do better by not restricting to coordinate subspaces?

Special case: $k = 1$

Orthogonal projection operator for 1-dimensional subspace W (i.e., a line):

$$\Pi_W \;=\; \vec{\alpha}\vec{\alpha}^{\mathsf{T}}$$

where $\vec{\alpha} \in \mathbb{R}^d$ is (any) unit vector in W

Special case: $k = 1$

Orthogonal projection operator for 1-dimensional subspace W (i.e., a line):

$$\Pi_W \;=\; \vec{\alpha}\vec{\alpha}^{\mathsf{T}}$$

where $\vec{\alpha} \in \mathbb{R}^d$ is (any) unit vector in W

Recall Pythagorean theorem:

$$\|\vec{X} - \vec{\mu}\|_2^2 \;=\; \|(\vec{X} - \vec{\mu}) - \Pi_W(\vec{X} - \vec{\mu})\|_2^2 + \|\Pi_W(\vec{X} - \vec{\mu})\|_2^2$$

So W that minimizes $\mathbb{E}\left[\|\vec{X} - \vec{\mu} - \Pi_W(\vec{X} - \vec{\mu})\|_2^2\right]$ is same as W that maximizes $\mathbb{E}\left[\|\Pi_W(\vec{X} - \vec{\mu})\|_2^2\right]$

For $k = 1$, this is same as finding unit vector $\vec{\alpha}$ that maximizes

$$\mathbb{E}\left[\|\vec{\alpha}\vec{\alpha}^{\mathsf{T}}(\vec{X} - \vec{\mu})\|_2^2\right] = \mathbb{E}\left[(\vec{\alpha}^{\mathsf{T}}(\vec{X} - \vec{\mu}))^2\right] = \mathrm{var}(\vec{\alpha} \cdot \vec{X})$$

Maximum variance linear function of a random vector

Question: What is the maximum variance achievable by a linear function of $\vec{X}$ given as a *unit vector*?

$$\max_{\substack{\vec{\alpha}\in\mathbb{R}^d \\ \text{s.t. } \|\vec{\alpha}\|_2=1}} \operatorname{var}(\vec{\alpha}\cdot\vec{X})$$

Maximum variance linear function of a random vector

Question: What is the maximum variance achievable by a linear function of $\vec{X}$ given as a *unit vector*?

$$\max_{\substack{\vec{\alpha}\in\mathbb{R}^d \\ \text{s.t. } \|\vec{\alpha}\|_2=1}} \operatorname{var}(\vec{\alpha}\cdot\vec{X})$$

- Answer: The largest eigenvalue of $\operatorname{cov}(\vec{X})$

Maximum variance linear function of a random vector

Question: What is the maximum variance achievable by a linear function of $\vec{X}$ given as a *unit vector*?

$$\max_{\substack{\vec{\alpha}\in\mathbb{R}^d \\ \text{s.t. } \|\vec{\alpha}\|_2=1}} \operatorname{var}(\vec{\alpha}\cdot\vec{X})$$

- Answer: The largest eigenvalue of $\operatorname{cov}(\vec{X})$

  Also called the **top eigenvalue** of $\operatorname{cov}(\vec{X})$

Maximum variance linear function of a random vector

Question: What is the maximum variance achievable by a linear function of $\vec{X}$ given as a *unit vector*?

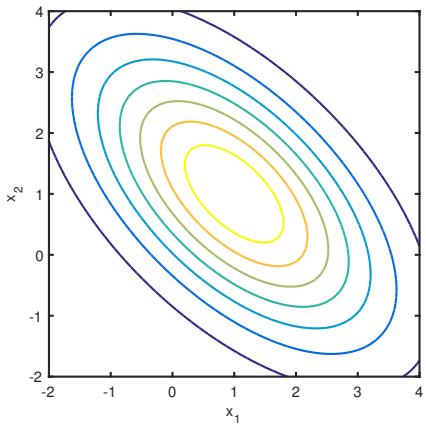
$$\max_{\substack{\vec{\alpha}\in\mathbb{R}^d \\ \text{s.t. } \|\vec{\alpha}\|_2=1}} \operatorname{var}(\vec{\alpha}\cdot\vec{X})$$

- Answer: The largest eigenvalue of $\operatorname{cov}(\vec{X})$

  Also called the **top eigenvalue** of $\operatorname{cov}(\vec{X})$

  Achieved by (unit length) eigenvector corresponding to that eigenvalue
  (a.k.a. **top eigenvector**)

Top eigenvector for covariance of a multivariate Gaussian



Top eigenvector direction defines the major axis of each ellipse

Optimality of top eigenvector

Let eigenvalues of $\mathrm{cov}(\vec{X})$ be $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_d \geq 0$, and let $\vec{v}_1, \vec{v}_2, \ldots, \vec{v}_d$ be corresponding orthonormal eigenvectors

Optimality of top eigenvector

Let eigenvalues of $\mathrm{cov}(\vec{X})$ be $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_d \geq 0$, and let $\vec{v}_1, \vec{v}_2, \ldots, \vec{v}_d$ be corresponding orthonormal eigenvectors

- Any vector $\vec{\alpha}$ can be written as

$$\vec{\alpha} \;=\; \sum_{i=1}^{d} (\vec{\alpha} \cdot \vec{v}_i)\, \vec{v}_i$$

Optimality of top eigenvector

Let eigenvalues of $\mathrm{cov}(\vec{X})$ be $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_d \geq 0$, and let $\vec{v}_1, \vec{v}_2, \ldots, \vec{v}_d$ be corresponding orthonormal eigenvectors

- Any vector $\vec{\alpha}$ can be written as

$$\vec{\alpha} \;=\; \sum_{i=1}^{d} (\vec{\alpha} \cdot \vec{v}_i)\, \vec{v}_i$$

Pythagorean theorem says $\sum_{i=1}^{d} (\vec{\alpha} \cdot \vec{v}_i)^2 = \|\vec{\alpha}\|_2^2$

Optimality of top eigenvector

Let eigenvalues of $\mathrm{cov}(\vec{X})$ be $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_d \geq 0$, and let $\vec{v}_1, \vec{v}_2, \ldots, \vec{v}_d$ be corresponding orthonormal eigenvectors

- Any vector $\vec{\alpha}$ can be written as

$$\vec{\alpha} \;=\; \sum_{i=1}^{d} (\vec{\alpha} \cdot \vec{v}_i)\, \vec{v}_i$$

  Pythagorean theorem says $\sum_{i=1}^{d} (\vec{\alpha} \cdot \vec{v}_i)^2 = \|\vec{\alpha}\|_2^2$

- Variance of $\vec{\alpha} \cdot \vec{X}$ for unit vector $\vec{\alpha}$:

$$\mathrm{var}(\vec{\alpha} \cdot \vec{X}) \quad = \quad \vec{\alpha}^{\mathsf{T}} \,\mathrm{cov}(\vec{X}) \vec{\alpha}$$

Optimality of top eigenvector

Let eigenvalues of $\mathrm{cov}(\vec{X})$ be $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_d \geq 0$, and let $\vec{v}_1, \vec{v}_2, \ldots, \vec{v}_d$ be corresponding orthonormal eigenvectors

- Any vector $\vec{\alpha}$ can be written as

$$\vec{\alpha} \;=\; \sum_{i=1}^{d} (\vec{\alpha} \cdot \vec{v}_i)\, \vec{v}_i$$

  Pythagorean theorem says $\sum_{i=1}^{d} (\vec{\alpha} \cdot \vec{v}_i)^2 = \|\vec{\alpha}\|_2^2$

- Variance of $\vec{\alpha} \cdot \vec{X}$ for unit vector $\vec{\alpha}$:

$$\begin{aligned} \mathrm{var}(\vec{\alpha} \cdot \vec{X}) &= \vec{\alpha}^{\mathsf{T}} \,\mathrm{cov}(\vec{X})\vec{\alpha} \\ &= \vec{\alpha}^{\mathsf{T}} \left(\sum_{i=1}^{d} \lambda_i \, \vec{v}_i \vec{v}_i^{\mathsf{T}} \right) \vec{\alpha} \end{aligned}$$

Optimality of top eigenvector

Let eigenvalues of $\mathrm{cov}(\vec{X})$ be $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_d \geq 0$, and let $\vec{v}_1, \vec{v}_2, \ldots, \vec{v}_d$ be corresponding orthonormal eigenvectors

- Any vector $\vec{\alpha}$ can be written as

$$\vec{\alpha} \;=\; \sum_{i=1}^{d} (\vec{\alpha} \cdot \vec{v}_i)\, \vec{v}_i$$

  Pythagorean theorem says $\sum_{i=1}^{d} (\vec{\alpha} \cdot \vec{v}_i)^2 = \|\vec{\alpha}\|_2^2$

- Variance of $\vec{\alpha} \cdot \vec{X}$ for unit vector $\vec{\alpha}$:

$$\begin{aligned} \mathrm{var}(\vec{\alpha} \cdot \vec{X}) &= \vec{\alpha}^{\mathsf{T}} \,\mathrm{cov}(\vec{X})\vec{\alpha} \\ &= \vec{\alpha}^{\mathsf{T}} \left(\sum_{i=1}^{d} \lambda_i \, \vec{v}_i \vec{v}_i^{\mathsf{T}} \right) \vec{\alpha} \\ &= \sum_{i=1}^{d} \lambda_i \, (\vec{\alpha} \cdot \vec{v}_i)^2 \end{aligned}$$

Optimality of top eigenvector

Let eigenvalues of $\mathrm{cov}(\vec{X})$ be $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_d \geq 0$, and let $\vec{v}_1, \vec{v}_2, \ldots, \vec{v}_d$ be corresponding orthonormal eigenvectors

- Any vector $\vec{\alpha}$ can be written as

$$\vec{\alpha} = \sum_{i=1}^{d} (\vec{\alpha} \cdot \vec{v}_i)\, \vec{v}_i$$

  Pythagorean theorem says $\sum_{i=1}^{d} (\vec{\alpha} \cdot \vec{v}_i)^2 = \|\vec{\alpha}\|_2^2$

- Variance of $\vec{\alpha} \cdot \vec{X}$ for unit vector $\vec{\alpha}$:

$$\begin{aligned} \mathrm{var}(\vec{\alpha} \cdot \vec{X}) &= \vec{\alpha}^{\mathsf{T}} \mathrm{cov}(\vec{X}) \vec{\alpha} \\ &= \vec{\alpha}^{\mathsf{T}} \left(\sum_{i=1}^{d} \lambda_i \, \vec{v}_i \vec{v}_i^{\mathsf{T}} \right) \vec{\alpha} \\ &= \sum_{i=1}^{d} \lambda_i \, (\vec{\alpha} \cdot \vec{v}_i)^2 \end{aligned}$$

- This is a "weighted" average of the λ_i's; maximized when all "weight" is put on λ_1

Optimality of top eigenvector

Let eigenvalues of $\mathrm{cov}(\vec{X})$ be $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_d \geq 0$, and let $\vec{v}_1, \vec{v}_2, \ldots, \vec{v}_d$ be corresponding orthonormal eigenvectors

- Any vector $\vec{\alpha}$ can be written as

$$\vec{\alpha} \;=\; \sum_{i=1}^{d} (\vec{\alpha} \cdot \vec{v}_i)\, \vec{v}_i$$

  Pythagorean theorem says $\sum_{i=1}^{d} (\vec{\alpha} \cdot \vec{v}_i)^2 = \|\vec{\alpha}\|_2^2$

- Variance of $\vec{\alpha} \cdot \vec{X}$ for unit vector $\vec{\alpha}$:

$$\begin{aligned} \mathrm{var}(\vec{\alpha} \cdot \vec{X}) &= \vec{\alpha}^{\mathsf{T}} \mathrm{cov}(\vec{X}) \vec{\alpha} \\ &= \vec{\alpha}^{\mathsf{T}} \left(\sum_{i=1}^{d} \lambda_i \, \vec{v}_i \vec{v}_i^{\mathsf{T}} \right) \vec{\alpha} \\ &= \sum_{i=1}^{d} \lambda_i \, (\vec{\alpha} \cdot \vec{v}_i)^2 \end{aligned}$$

- This is a "weighted" average of the λ_i's; maximized when all "weight" is put on λ_1
- To put all "weight" on λ_1, need $(\vec{\alpha} \cdot \vec{v}_1)^2 = 1$; achieved by $\vec{\alpha} = \pm \vec{v}_1$

Top eigen{value,vector} on MNIST data

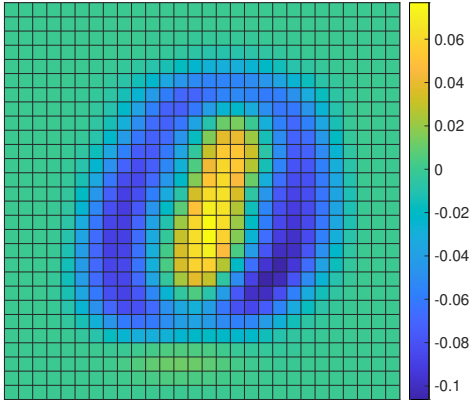
Top eigenvalue:

$$\lambda_1 \approx 0.1 \times \mathbb{E}\|\vec{X} - \vec{\mu}\|_2^2$$

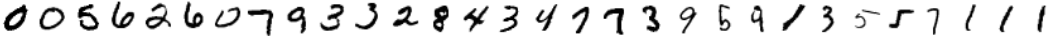
Expected reconstruction error:

$$\approx 0.9 \times \mathbb{E}\|\vec{X} - \vec{\mu}\|_2^2$$

Top eigenvector:



Some MNIST digits sorted by projection to span of top eigenvector:



General case: $k \geq 1$

Orthogonal projection operator for k-dimensional subspace W:

$$\Pi_W = \sum_{i=1}^{k} \vec{\alpha}_i \vec{\alpha}_i^{\mathsf{T}}$$

where $\vec{\alpha}_1, \ldots, \vec{\alpha}_k \in \mathbb{R}^d$ is (any) orthonormal basis for W

General case: $k \geq 1$

Orthogonal projection operator for k-dimensional subspace W:

$$\Pi_W \;=\; \sum_{i=1}^{k} \vec{\alpha}_i \vec{\alpha}_i^{\mathsf{T}}$$

where $\vec{\alpha}_1, \ldots, \vec{\alpha}_k \in \mathbb{R}^d$ is (any) orthonormal basis for W

Recall Pythagorean theorem:

$$\|\vec{X} - \vec{\mu}\|_2^2 \;=\; \|(\vec{X} - \vec{\mu}) - \Pi_W(\vec{X} - \vec{\mu})\|_2^2 + \|\Pi_W(\vec{X} - \vec{\mu})\|_2^2$$

So W that minimizes $\mathbb{E}\left[\|\vec{X} - \vec{\mu} - \Pi_W(\vec{X} - \vec{\mu})\|_2^2\right]$ is same as W that maximizes $\mathbb{E}\left[\|\Pi_W(\vec{X} - \vec{\mu})\|_2^2\right]$

This is same as finding orthonormal $\vec{\alpha}_1, \ldots, \vec{\alpha}_k$ that maximizes

$$\mathbb{E}\left[\left\|\sum_{i=1}^{k} \vec{\alpha}_i \vec{\alpha}_i^{\mathsf{T}}(\vec{X} - \vec{\mu})\right\|_2^2\right] = \mathbb{E}\left[\sum_{i=1}^{k} (\vec{\alpha}_i^{\mathsf{T}}(\vec{X} - \vec{\mu}))^2\right] = \sum_{i=1}^{k} \mathrm{var}(\vec{\alpha}_i \cdot \vec{X})$$

Optimality of the top k eigenvectors

Question: What's the maximum variance "accounted for" by k dimensional subspace?

$$\max_{\substack{\vec{\alpha}_1,\ldots,\vec{\alpha}_k \in \mathbb{R}^d \\ \text{orthonormal}}} \sum_{i=1}^{k} \text{var}(\vec{\alpha}_i \cdot \vec{X})$$

Optimality of the top k eigenvectors

Question: What's the maximum variance "accounted for" by k dimensional subspace?

$$\max_{\substack{\vec{\alpha}_1,\ldots,\vec{\alpha}_k \in \mathbb{R}^d \\ \text{orthonormal}}} \sum_{i=1}^{k} \operatorname{var}(\vec{\alpha}_i \cdot \vec{X})$$

- Answer: Sum of k largest eigenvalues of $\operatorname{cov}(\vec{X})$
  (sum of the **top k eigenvalues**)

Optimality of the top k eigenvectors

Question: What's the maximum variance "accounted for" by k dimensional subspace?

$$\max_{\substack{\vec{\alpha}_1,\ldots,\vec{\alpha}_k \in \mathbb{R}^d \\ \text{orthonormal}}} \sum_{i=1}^{k} \operatorname{var}(\vec{\alpha}_i \cdot \vec{X})$$

- Answer: Sum of k largest eigenvalues of $\operatorname{cov}(\vec{X})$
  (sum of the **top k eigenvalues**)

  Achieved by any orthonormal basis for the span of k corresponding eigenvectors
  (span of **top k eigenvectors**, a.k.a. **top k eigenspace**)

Optimality of the top k eigenvectors

Question: What's the maximum variance "accounted for" by k dimensional subspace?

$$\max_{\substack{\vec{\alpha}_1,\ldots,\vec{\alpha}_k \in \mathbb{R}^d \\ \text{orthonormal}}} \sum_{i=1}^{k} \operatorname{var}(\vec{\alpha}_i \cdot \vec{X})$$

- Answer: Sum of k largest eigenvalues of $\operatorname{cov}(\vec{X})$
  (sum of the **top k eigenvalues**)

  Achieved by any orthonormal basis for the span of k corresponding eigenvectors
  (span of **top k eigenvectors**, a.k.a. **top k eigenspace**)

If $\vec{v}_1, \ldots, \vec{v}_k$ are top k eigenvectors of $\operatorname{cov}(\vec{X})$, we create k "eigenfeatures" $\vec{v}_i \cdot \vec{X}$ for $i = 1, \ldots, k$

Optimality of the top k eigenvectors

Question: What's the maximum variance "accounted for" by k dimensional subspace?

$$\max_{\substack{\vec{\alpha}_1,\ldots,\vec{\alpha}_k \in \mathbb{R}^d \\ \text{orthonormal}}} \sum_{i=1}^{k} \operatorname{var}(\vec{\alpha}_i \cdot \vec{X})$$

- Answer: Sum of k largest eigenvalues of $\operatorname{cov}(\vec{X})$
  (sum of the **top k eigenvalues**)

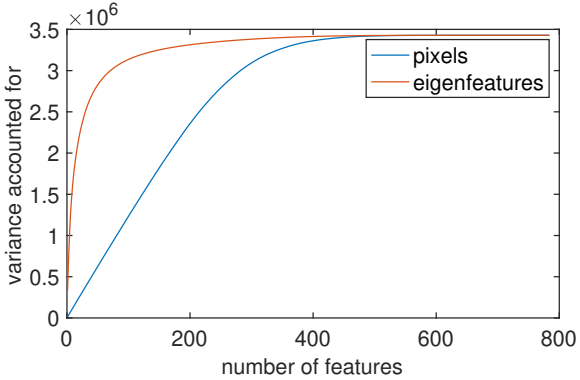
  Achieved by any orthonormal basis for the span of k corresponding eigenvectors
  (span of **top k eigenvectors**, a.k.a. **top k eigenspace**)

If $\vec{v}_1, \ldots, \vec{v}_k$ are top k eigenvectors of $\operatorname{cov}(\vec{X})$, we create k "eigenfeatures" $\vec{v}_i \cdot \vec{X}$ for $i = 1, \ldots, k$

Vectors $\vec{v}_1, \ldots, \vec{v}_k$ are called the **top k principal components** of (the distribution of) $\vec{X}$
Computing these "eigenfeatures" is called **(k-dim.) Principal Components Analysis (PCA)**

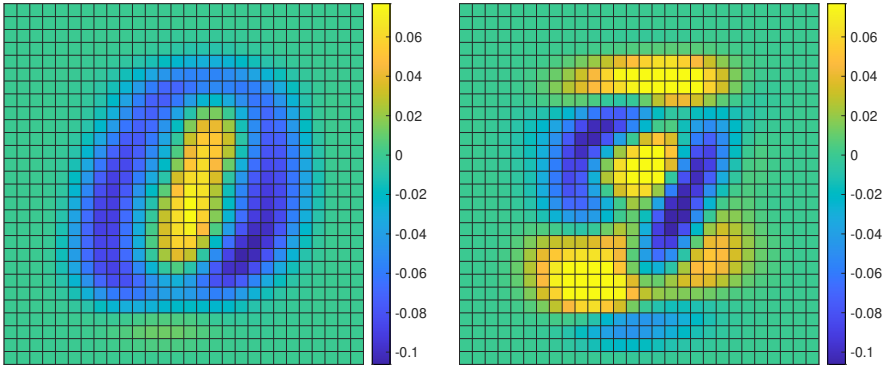
Variance accounted for on MNIST data

MNIST data: Need $k = 87$ "eigenfeatures" to reduce expected reconstruction error to $0.1 \times \mathbb{E}\|\vec{X} - \vec{\mu}\|_2^2$



("Variance accounted for" $= \sum_{i=1}^{k} \mathrm{var}(\vec{\alpha}_i \cdot \vec{X})$)

MNIST eigenvectors



$\vec{v}_1$ $\vec{v}_2$

Orthogonality: $\vec{v}_1 \cdot \vec{v}_2 = 0$

New features from PCA

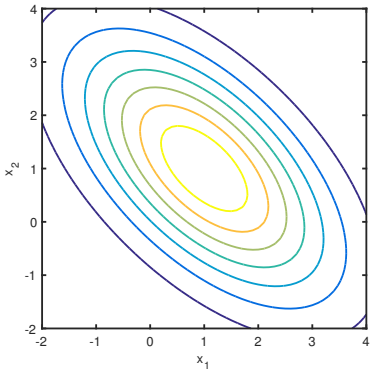
Properties of eigenfeatures $\vec{v}_i \cdot \vec{X}$ for $i = 1, \ldots, k$:

- $\mathrm{var}(\vec{v}_i \cdot \vec{X}) = \lambda_i$ for all i
- $\mathrm{cov}(\vec{v}_i \cdot \vec{X}, \vec{v}_j \cdot \vec{X}) = 0$ for $i \neq j$

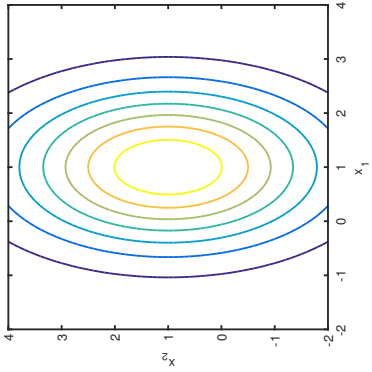
New features from PCA

Properties of eigenfeatures $\vec{v}_i \cdot \vec{X}$ for $i = 1, \ldots, k$:

- $\mathrm{var}(\vec{v}_i \cdot \vec{X}) = \lambda_i$ for all i
- $\mathrm{cov}(\vec{v}_i \cdot \vec{X}, \vec{v}_j \cdot \vec{X}) = 0$ for $i \neq j$



Original features



Eigenfeatures

PCA and beyond

PCA and beyond

- Major use of PCA: **Dimension reduction**
  Map data points in $\mathbb{R}^d$ to $\mathbb{R}^k$ for $k < d$
  (When $k = 2$ or $k = 3$, useful for visualization, maybe!)

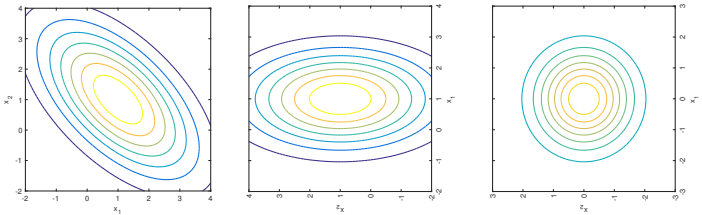
PCA and beyond

- Major use of PCA: **Dimension reduction**
  Map data points in $\mathbb{R}^d$ to $\mathbb{R}^k$ for $k < d$
  (When $k = 2$ or $k = 3$, useful for visualization, maybe!)
- k-dim. PCA + OLS = **Principal components regression (PCR)**
  Different inductive bias compared to ridge regression!

PCA and beyond

- Major use of PCA: **Dimension reduction**
  Map data points in $\mathbb{R}^d$ to $\mathbb{R}^k$ for $k < d$
  (When $k = 2$ or $k = 3$, useful for visualization, maybe!)
- k-dim. PCA + OLS = **Principal components regression (PCR)**
  Different inductive bias compared to ridge regression!
- **Whitening**: Transforming to isotropy

$$\frac{1}{\sqrt{\lambda_i}} \vec{v}_i \cdot \vec{X} \quad \text{for } i = 1, \ldots, k$$



Original features | Eigenfeatures | Whitened features

Postscript: "Un-centered" PCA

Everything about eigen{values,vectors} of $\mathrm{cov}(\vec{X})$ also applies to **second-moment matrix** $\mathbb{E}[\vec{X}\vec{X}^{\mathsf{T}}]$, ... with following changes:

- Replace $\mathrm{var}(\vec{\alpha} \cdot \vec{X})$ with $\mathbb{E}[(\vec{\alpha} \cdot \vec{X})^2]$
- Change "expected reconstruction error" to $\mathbb{E}\left[\|\vec{X} - \Pi^{\mathsf{T}}\vec{X}\|_2^2\right]$

Postscript: "Un-centered" PCA

Everything about eigen{values,vectors} of $\mathrm{cov}(\vec{X})$ also applies to **second-moment matrix** $\mathbb{E}[\vec{X}\vec{X}^\mathsf{T}]$, ... with following changes:

- Replace $\mathrm{var}(\vec{\alpha} \cdot \vec{X})$ with $\mathbb{E}[(\vec{\alpha} \cdot \vec{X})^2]$
- Change "expected reconstruction error" to $\mathbb{E}\left[\|\vec{X} - \Pi^\mathsf{T}\vec{X}\|_2^2\right]$

If $A = [\vec{x}_1|\cdots|\vec{x}_n]^\mathsf{T} \in \mathbb{R}^{n\times d}$, then (empirical) second-moment matrix is

$$\frac{1}{n}A^\mathsf{T}A$$

Postscript: "Un-centered" PCA

Everything about eigen{values,vectors} of $\mathrm{cov}(\vec{X})$ also applies to **second-moment matrix** $\mathbb{E}[\vec{X}\vec{X}^\mathsf{T}]$, ... with following changes:

- Replace $\mathrm{var}(\vec{\alpha} \cdot \vec{X})$ with $\mathbb{E}[(\vec{\alpha} \cdot \vec{X})^2]$
- Change "expected reconstruction error" to $\mathbb{E}\left[\|\vec{X} - \Pi^\mathsf{T}\vec{X}\|_2^2\right]$

If $A = [\vec{x}_1 | \cdots | \vec{x}_n]^\mathsf{T} \in \mathbb{R}^{n \times d}$, then (empirical) second-moment matrix is

$$\frac{1}{n} A^\mathsf{T} A$$

Question: How do eigen{values,vectors} of $A^\mathsf{T}A$ relate to eigen{values,vectors} of $\frac{1}{n}A^\mathsf{T}A$?

Postscript: "Un-centered" PCA

Everything about eigen{values,vectors} of $\mathrm{cov}(\vec{X})$ also applies to **second-moment matrix** $\mathbb{E}[\vec{X}\vec{X}^\mathsf{T}]$, ... with following changes:

- Replace $\mathrm{var}(\vec{\alpha} \cdot \vec{X})$ with $\mathbb{E}[(\vec{\alpha} \cdot \vec{X})^2]$
- Change "expected reconstruction error" to $\mathbb{E}\left[\|\vec{X} - \Pi^\mathsf{T}\vec{X}\|_2^2\right]$

If $A = [\vec{x}_1 | \cdots | \vec{x}_n]^\mathsf{T} \in \mathbb{R}^{n \times d}$, then (empirical) second-moment matrix is

$$\frac{1}{n} A^\mathsf{T} A$$

Question: How do eigen{values,vectors} of $A^\mathsf{T}A$ relate to eigen{values,vectors} of $\frac{1}{n}A^\mathsf{T}A$?

Question: How do eigen{values,vectors} of $A^\mathsf{T}A$ relate to eigen{values,vectors} of A?

Eigenvector computation

Simple approach to top eigenvector computation

Let eigendecomposition of $A^\mathsf{T}A$ be $\sum_{i=1}^{d} \lambda_i \, \vec{v}_i \, \vec{v}_i^\mathsf{T}$

Question: How to compute top eigenvector $\vec{v}_1$ of $A^\mathsf{T}A$?

Simple approach to top eigenvector computation

Let eigendecomposition of $A^\mathsf{T}A$ be $\sum_{i=1}^{d} \lambda_i \, \vec{v}_i \, \vec{v}_i^\mathsf{T}$

Question: How to compute top eigenvector $\vec{v}_1$ of $A^\mathsf{T}A$?

- **Fact.** For any integer t, $(A^\mathsf{T}A)^t = \sum_{i=1}^{d} \lambda_i^t \, \vec{v}_i \, \vec{v}_i^\mathsf{T}$

Simple approach to top eigenvector computation

Let eigendecomposition of $A^\top A$ be $\sum_{i=1}^{d} \lambda_i \, \vec{v}_i \, \vec{v}_i^\top$

Question: How to compute top eigenvector $\vec{v}_1$ of $A^\top A$?

- **Fact.** For any integer t, $(A^\top A)^t = \sum_{i=1}^{d} \lambda_i^t \, \vec{v}_i \, \vec{v}_i^\top$

If t is large, and $\lambda_1 > \lambda_2$, then

$$(A^\top A)^t \;=\; \lambda_1^t \left(\vec{v}_1 \, \vec{v}_1^\top + \sum_{i=2}^{d} \left(\frac{\lambda_i}{\lambda_1} \right)^t \vec{v}_i \, \vec{v}_i^\top \right) \;\approx\; \lambda_1^t \, \vec{v}_1 \, \vec{v}_1^\top$$

where $\approx$ holds when t is sufficiently large

Simple approach to top eigenvector computation

Let eigendecomposition of $A^\mathsf{T}A$ be $\sum_{i=1}^{d} \lambda_i \, \vec{v}_i \, \vec{v}_i^\mathsf{T}$

Question: How to compute top eigenvector $\vec{v}_1$ of $A^\mathsf{T}A$?

- **Fact.** For any integer t, $(A^\mathsf{T}A)^t = \sum_{i=1}^{d} \lambda_i^t \, \vec{v}_i \, \vec{v}_i^\mathsf{T}$

If t is large, and $\lambda_1 > \lambda_2$, then

$$(A^\mathsf{T}A)^t \;=\; \lambda_1^t \left(\vec{v}_1 \, \vec{v}_1^\mathsf{T} + \sum_{i=2}^{d} \left(\frac{\lambda_i}{\lambda_1} \right)^t \vec{v}_i \, \vec{v}_i^\mathsf{T} \right) \;\approx\; \lambda_1^t \, \vec{v}_1 \, \vec{v}_1^\mathsf{T}$$

where $\approx$ holds when t is sufficiently large

Main idea:
If t is sufficiently large, multiplying $(A^\mathsf{T}A)^t$ by (almost) any vector yields a vector that is (approximately) in span of $\vec{v}_1$

Power iteration

Slow implementation of main idea: First compute $(A^\top A)^t$, and then multiply by a vector

Power iteration

Slow implementation of main idea: First compute $(A^\top A)^t$, and then multiply by a vector

Better implementation:

Power iteration (a.k.a. power method)

- Initialize $\vec{x}^{(0)} \in \mathbb{R}^d$ randomly (e.g., draw from $\mathrm{N}(\vec{0}, I_d)$)
- For $t = 1, 2, \ldots$ until stopping condition is satisfied:

$$\begin{aligned} \vec{y} &:= A\vec{x}^{(t-1)} \\ \vec{z} &:= A^\top \vec{y} \\ \vec{x}^{(t)} &:= \vec{z}/\|\vec{z}\|_2 \end{aligned}$$

Power iteration

Slow implementation of main idea: First compute $(A^\mathsf{T}A)^t$, and then multiply by a vector

Better implementation:

Power iteration (a.k.a. power method)

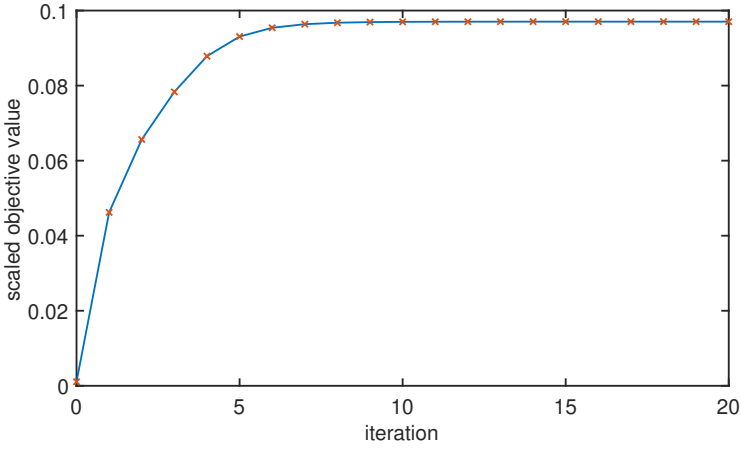
- Initialize $\vec{x}^{(0)} \in \mathbb{R}^d$ randomly (e.g., draw from $\mathrm{N}(\vec{0}, I_d)$)
- For $t = 1, 2, \ldots$ until stopping condition is satisfied:

$$\begin{aligned} \vec{y} &:= A\vec{x}^{(t-1)} \\ \vec{z} &:= A^\mathsf{T}\vec{y} \\ \vec{x}^{(t)} &:= \vec{z}/\|\vec{z}\|_2 \end{aligned}$$

After t iterations, this computes a scaling of $(A^\mathsf{T}A)^t\vec{x}^{(0)}$
(Scaling is due to normalization steps "$\vec{x}^{(t)} := \vec{z}/\|\vec{z}\|_2$")

Example: Power iteration on centered MNIST data

Scaled objective: $J(\vec{x}) = \dfrac{1}{\mathrm{tr}(A^\mathsf{T}A)} \cdot \dfrac{\vec{x}^\mathsf{T}(A^\mathsf{T}A)\vec{x}}{\|\vec{x}\|_2^2}$



Some issues with power iteration

- "Issue" 1: If $\lambda_1 = \lambda_2$, then $\vec{v}_1$ is not uniquely defined

Some issues with power iteration

- "Issue" 1: If $\lambda_1 = \lambda_2$, then $\vec{v}_1$ is not uniquely defined
- Issue 2: If $\lambda_1 > \lambda_2$ but very close, may need extremely large t

Some issues with power iteration

- "Issue" 1: If $\lambda_1 = \lambda_2$, then $\vec{v}_1$ is not uniquely defined
- Issue 2: If $\lambda_1 > \lambda_2$ but very close, may need extremely large t
- Issue 3: What about λ_1?

Some issues with power iteration

- "Issue" 1: If $\lambda_1 = \lambda_2$, then $\vec{v}_1$ is not uniquely defined
- Issue 2: If $\lambda_1 > \lambda_2$ but very close, may need extremely large t
- Issue 3: What about λ_1?
  $\|A\vec{v}_1\|_2^2 = \lambda_1$

Some issues with power iteration

- "Issue" 1: If $\lambda_1 = \lambda_2$, then $\vec{v}_1$ is not uniquely defined
- Issue 2: If $\lambda_1 > \lambda_2$ but very close, may need extremely large t
- Issue 3: What about λ_1?
  $\|A\vec{v}_1\|_2^2 = \lambda_1$
- Issue 4: What about $\vec{v}_2, \vec{v}_3, \ldots$ and $\lambda_2, \lambda_3, \ldots$?

Some issues with power iteration

- "Issue" 1: If $\lambda_1 = \lambda_2$, then $\vec{v}_1$ is not uniquely defined
- Issue 2: If $\lambda_1 > \lambda_2$ but very close, may need extremely large t
- Issue 3: What about λ_1?
  $\|A\vec{v}_1\|_2^2 = \lambda_1$
- Issue 4: What about $\vec{v}_2, \vec{v}_3, \ldots$ and $\lambda_2, \lambda_3, \ldots$?
  Can use **deflation**

Some issues with power iteration

- "Issue" 1: If $\lambda_1 = \lambda_2$, then $\vec{v}_1$ is not uniquely defined
- Issue 2: If $\lambda_1 > \lambda_2$ but very close, may need extremely large t
- Issue 3: What about λ_1?
  $\|A\vec{v}_1\|_2^2 = \lambda_1$
- Issue 4: What about $\vec{v}_2, \vec{v}_3, \ldots$ and $\lambda_2, \lambda_3, \ldots$?
  Can use **deflation**
  If you only care about subspace spanned by $\vec{v}_1, \ldots, \vec{v}_k$: use **subspace iteration**

Low-rank Approximation

Low-rank matrix approximation

Given data matrix $A = [\vec{x}_1 | \cdots | \vec{x}_n]^\mathsf{T} \in \mathbb{R}^{n \times d}$, how can we best approximate it by the product BC for some $B \in \mathbb{R}^{n \times k}$ and $C \in \mathbb{R}^{k \times d}$?

Low-rank matrix approximation

Given data matrix $A = [\vec{x}_1 | \cdots | \vec{x}_n]^\mathsf{T} \in \mathbb{R}^{n \times d}$, how can we best approximate it by the product BC for some $B \in \mathbb{R}^{n \times k}$ and $C \in \mathbb{R}^{k \times d}$?

- "Low-rank" since $\operatorname{rank}(BC) \leq k$

Low-rank matrix approximation

Given data matrix $A = [\vec{x}_1 | \cdots | \vec{x}_n]^\mathsf{T} \in \mathbb{R}^{n \times d}$, how can we best approximate it by the product BC for some $B \in \mathbb{R}^{n \times k}$ and $C \in \mathbb{R}^{k \times d}$?

- "Low-rank" since $\operatorname{rank}(BC) \leq k$
- $k \leq \min\{n, d\}$ (typically $k \ll \min\{n, d\}$)

Low-rank matrix approximation

Given data matrix $A = [\vec{x}_1 | \cdots | \vec{x}_n]^\mathsf{T} \in \mathbb{R}^{n \times d}$, how can we best approximate it by the product BC for some $B \in \mathbb{R}^{n \times k}$ and $C \in \mathbb{R}^{k \times d}$?

- "Low-rank" since $\mathrm{rank}(BC) \leq k$
- $k \leq \min\{n, d\}$ (typically $k \ll \min\{n, d\}$)
- Measure of approximation error: **Frobenius norm**

$$J(B, C) = \|A - BC\|_\mathrm{F}^2$$

where

$$\|M\|_\mathrm{F} = \sqrt{\sum_{i=1}^{n} \sum_{j=1}^{d} M_{i,j}^2}$$

treats $n \times d$ matrix M as vector in $\mathbb{R}^{nd}$

Low-rank matrix approximation

Given data matrix $A = [\vec{x}_1 | \cdots | \vec{x}_n]^\mathsf{T} \in \mathbb{R}^{n \times d}$, how can we best approximate it by the product BC for some $B \in \mathbb{R}^{n \times k}$ and $C \in \mathbb{R}^{k \times d}$?

- "Low-rank" since $\mathrm{rank}(BC) \leq k$
- $k \leq \min\{n, d\}$ (typically $k \ll \min\{n, d\}$)
- Measure of approximation error: **Frobenius norm**

$$J(B, C) = \|A - BC\|_\mathrm{F}^2$$

where

$$\|M\|_\mathrm{F} = \sqrt{\sum_{i=1}^{n} \sum_{j=1}^{d} M_{i,j}^2}$$

treats $n \times d$ matrix M as vector in $\mathbb{R}^{nd}$

- If B is fixed, then this looks like the SSE objective from OLS!

Low-rank matrix approximation

Given data matrix $A = [\vec{x}_1 | \cdots | \vec{x}_n]^\mathsf{T} \in \mathbb{R}^{n \times d}$, how can we best approximate it by the product BC for some $B \in \mathbb{R}^{n \times k}$ and $C \in \mathbb{R}^{k \times d}$?

- "Low-rank" since $\mathrm{rank}(BC) \leq k$
- $k \leq \min\{n, d\}$ (typically $k \ll \min\{n, d\}$)
- Measure of approximation error: **Frobenius norm**

$$J(B, C) = \|A - BC\|_\mathrm{F}^2$$

where

$$\|M\|_\mathrm{F} = \sqrt{\sum_{i=1}^{n} \sum_{j=1}^{d} M_{i,j}^2}$$

treats $n \times d$ matrix M as vector in $\mathbb{R}^{nd}$

- If B is fixed, then this looks like the SSE objective from OLS!
- Same if C is fixed.

Low-rank matrix approximation

Given data matrix $A = [\vec{x}_1 | \cdots | \vec{x}_n]^\mathsf{T} \in \mathbb{R}^{n \times d}$, how can we best approximate it by the product BC for some $B \in \mathbb{R}^{n \times k}$ and $C \in \mathbb{R}^{k \times d}$?

- "Low-rank" since $\mathrm{rank}(BC) \leq k$
- $k \leq \min\{n, d\}$ (typically $k \ll \min\{n, d\}$)
- Measure of approximation error: **Frobenius norm**

$$J(B, C) = \|A - BC\|_\mathrm{F}^2$$

where

$$\|M\|_\mathrm{F} = \sqrt{\sum_{i=1}^{n} \sum_{j=1}^{d} M_{i,j}^2}$$

treats $n \times d$ matrix M as vector in $\mathbb{R}^{nd}$

- If B is fixed, then this looks like the SSE objective from OLS!
- Same if C is fixed.
- But apparently more complicated, since both B and C are optimization variables! (E.g., J is not a convex function of (B, C))

Two optimization methods

Gradient descent:

- Start with initial values $B^{(0)} \in \mathbb{R}^{n \times k}$ and $C^{(0)} \in \mathbb{R}^{k \times d}$
- For iteration $t = 1, 2, \ldots$ until stopping condition is satisfied:

$$B^{(t)} := B^{(t-1)} - \eta \frac{\partial J}{\partial B}(B^{(t-1)}, C^{(t-1)})$$
$$C^{(t)} := C^{(t-1)} - \eta \frac{\partial J}{\partial C}(B^{(t-1)}, C^{(t-1)})$$

Two optimization methods

Gradient descent:

- Start with initial values $B^{(0)} \in \mathbb{R}^{n \times k}$ and $C^{(0)} \in \mathbb{R}^{k \times d}$
- For iteration $t = 1, 2, \ldots$ until stopping condition is satisfied:

$$B^{(t)} := B^{(t-1)} - \eta \frac{\partial J}{\partial B}(B^{(t-1)}, C^{(t-1)})$$
$$C^{(t)} := C^{(t-1)} - \eta \frac{\partial J}{\partial C}(B^{(t-1)}, C^{(t-1)})$$

Alternating minimization:

- Start with initial values $B^{(0)} \in \mathbb{R}^{n \times k}$ and $C^{(0)} \in \mathbb{R}^{k \times d}$
- For iteration $t = 1, 2, \ldots$ until stopping condition is satisfied:

$$B^{(t)} := \operatorname*{arg\,min}_{B \in \mathbb{R}^{n \times k}} J(B, C^{(t-1)})$$
$$C^{(t)} := \operatorname*{arg\,min}_{C \in \mathbb{R}^{k \times d}} J(B^{(t)}, C)$$

Singular value decomposition

Theorem (Schmidt, 1907; Eckart & Young, 1936)**.** Optimal solution to the low-rank matrix approximation problem is given by truncating the *singular value decomposition* of A

Singular value decomposition

Theorem (Schmidt, 1907; Eckart & Young, 1936)**.** Optimal solution to the low-rank matrix approximation problem is given by truncating the *singular value decomposition* of A

Singular value decomposition (SVD) of A: Every matrix $A \in \mathbb{R}^{n \times d}$ (with $r := \operatorname{rank}(A)$) can be written as

$$A = \sum_{i=1}^{r} s_i \, \vec{u}_i \, \vec{v}_i^{\mathsf{T}}$$

for some $\{(s_i, \vec{u}_i, \vec{v}_i)\}_{i=1}^{r}$, which satisfy

- $s_1 \geq \cdots \geq s_r > 0$ (**singular values** of A; sorting this way is by convention)
- $\vec{u}_1, \ldots, \vec{u}_r \in \mathbb{R}^n$ are orthonormal (**left singular vectors** of A)
- $\vec{v}_1, \ldots, \vec{v}_r \in \mathbb{R}^d$ are orthonormal (**right singular vectors** of A)

Singular value decomposition

Theorem (Schmidt, 1907; Eckart & Young, 1936)**.** Optimal solution to the low-rank matrix approximation problem is given by truncating the *singular value decomposition* of A

Singular value decomposition (SVD) of A: Every matrix $A \in \mathbb{R}^{n \times d}$ (with $r := \operatorname{rank}(A)$) can be written as

$$A = \sum_{i=1}^{r} s_i \, \vec{u}_i \, \vec{v}_i^{\mathsf{T}}$$

for some $\{(s_i, \vec{u}_i, \vec{v}_i)\}_{i=1}^{r}$, which satisfy

- $s_1 \geq \cdots \geq s_r > 0$ (**singular values** of A; sorting this way is by convention)
- $\vec{u}_1, \ldots, \vec{u}_r \in \mathbb{R}^n$ are orthonormal (**left singular vectors** of A)
- $\vec{v}_1, \ldots, \vec{v}_r \in \mathbb{R}^d$ are orthonormal (**right singular vectors** of A)

Can also write as

$$A = USV^{\mathsf{T}}$$

where $U := [\vec{u}_1 | \cdots | \vec{u}_r] \in \mathbb{R}^{n \times r}$, $S := \operatorname{diag}(s_1, \ldots, s_r) \in \mathbb{R}^{r \times r}$, and $V := [\vec{v}_1 | \cdots | \vec{v}_r] \in \mathbb{R}^{d \times r}$

Truncated SVD

Suppose SVD of A is given by

$$A = \sum_{i=1}^{\text{rank}(A)} s_i \, \vec{u}_i \, \vec{v}_i^{\mathsf{T}}$$

Truncated SVD

Suppose SVD of A is given by

$$A = \sum_{i=1}^{\operatorname{rank}(A)} s_i \, \vec{u}_i \, \vec{v}_i^{\mathsf{T}}$$

Theorem (Schmidt, 1907; Eckart & Young, 1936)**.** The **rank-k SVD** of A (for $k \leq \operatorname{rank}(A)$),

$$A_k := \sum_{i=1}^{k} s_i \, \vec{u}_i \, \vec{v}_i^{\mathsf{T}},$$

satisfies

$$\|A - A_k\|_{\mathrm{F}}^2 \;=\; \min_{\substack{M \in \mathbb{R}^{n \times d} \\ \text{s.t. } \operatorname{rank}(M) \leq k}} \|A - M\|_{\mathrm{F}}^2 \;=\; \sum_{i=k+1}^{\operatorname{rank}(A)} s_i^2$$

Truncated SVD

Suppose SVD of A is given by

$$A = \sum_{i=1}^{\operatorname{rank}(A)} s_i \, \vec{u}_i \, \vec{v}_i^{\mathsf{T}}$$

Theorem (Schmidt, 1907; Eckart & Young, 1936)**.** The **rank-k SVD** of A (for $k \leq \operatorname{rank}(A)$),

$$A_k := \sum_{i=1}^{k} s_i \, \vec{u}_i \, \vec{v}_i^{\mathsf{T}},$$

satisfies

$$\|A - A_k\|_{\mathrm{F}}^2 \;=\; \min_{\substack{M \in \mathbb{R}^{n \times d} \\ \text{s.t. } \operatorname{rank}(M) \leq k}} \|A - M\|_{\mathrm{F}}^2 \;=\; \sum_{i=k+1}^{\operatorname{rank}(A)} s_i^2$$

Can also write $A_k = U_k S_k V_k^{\mathsf{T}}$ where $U_k := [\vec{u}_1 | \cdots | \vec{u}_k] \in \mathbb{R}^{n \times k}$, $S_k := \operatorname{diag}(s_1, \ldots, s_k) \in \mathbb{R}^{k \times k}$, and $V_k := [\vec{v}_1 | \cdots | \vec{v}_k] \in \mathbb{R}^{d \times k}$

Connection to PCA

Observation:

$$A^\mathsf{T}A \;=\; (VSU^\mathsf{T})(USV^\mathsf{T}) \;=\; VS^2V^\mathsf{T} \;=\; \sum_{i=1}^{r} s_i^2\,\vec{v}_i\,\vec{v}_i^\mathsf{T}$$

- Non-zero eigenvalues of $A^\mathsf{T}A$ are squares of singular values of A
- Corresponding eigenvectors of $A^\mathsf{T}A$ are right singular vectors of A

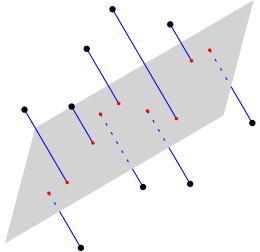
Connection to PCA

Observation:

$$A^\mathsf{T}A \;=\; (VSU^\mathsf{T})(USV^\mathsf{T}) \;=\; VS^2V^\mathsf{T} \;=\; \sum_{i=1}^{r} s_i^2\,\vec{v}_i\,\vec{v}_i^\mathsf{T}$$

- Non-zero eigenvalues of $A^\mathsf{T}A$ are squares of singular values of A
- Corresponding eigenvectors of $A^\mathsf{T}A$ are right singular vectors of A

Computing rank-k SVD is same as applying (un-centered) dim. k PCA feature mapping!



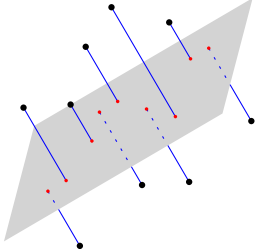
Connection to PCA

Observation:

$$A^\mathsf{T} A \;=\; (VSU^\mathsf{T})(USV^\mathsf{T}) \;=\; VS^2V^\mathsf{T} \;=\; \sum_{i=1}^{r} s_i^2\, \vec{v}_i\, \vec{v}_i^\mathsf{T}$$

- Non-zero eigenvalues of $A^\mathsf{T}A$ are squares of singular values of A
- Corresponding eigenvectors of $A^\mathsf{T}A$ are right singular vectors of A

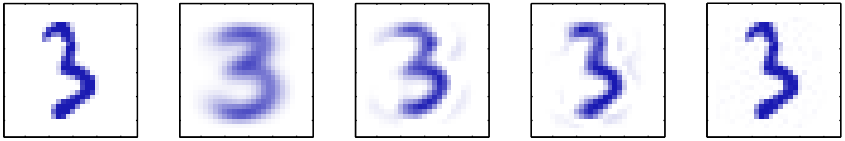
Computing rank-k SVD is same as applying (un-centered) dim. k PCA feature mapping!



Rows of $U_k S_k$ are the k-dimensional vectors of "eigenfeatures"

MNIST data example

MNIST data example: Compare original image to corresponding row of A_k ($k \in \{1, 10, 50, 200\}$)



Application: Latent Semantic Analysis

Document modeling

Suppose you have corpus of n text documents

- Each document represented using "bag-of-words" count vectors
- $d =$ vocabulary size

$$A := \begin{array}{c|cccc} & \text{aardvark} & \text{abacus} & \text{abalone} & \cdots \\ \hline \text{doc 1} & 3 & 0 & 0 & \cdots \\ \text{doc 2} & 7 & 0 & 4 & \cdots \\ \text{doc 3} & 0 & 4 & 0 & \cdots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{array}$$

Document modeling

Suppose you have corpus of n text documents

- Each document represented using "bag-of-words" count vectors
- $d =$ vocabulary size

$$A := \begin{array}{c|cccc} & \text{aardvark} & \text{abacus} & \text{abalone} & \cdots \\ \hline \text{doc 1} & 3 & 0 & 0 & \cdots \\ \text{doc 2} & 7 & 0 & 4 & \cdots \\ \text{doc 3} & 0 & 4 & 0 & \cdots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{array}$$

Question: How to find documents related to mathematics?

Document modeling

Suppose you have corpus of n text documents

- Each document represented using "bag-of-words" count vectors
- $d =$ vocabulary size

$$A := \begin{array}{c|cccc} & \text{aardvark} & \text{abacus} & \text{abalone} & \cdots \\ \hline \text{doc 1} & 3 & 0 & 0 & \cdots \\ \text{doc 2} & 7 & 0 & 4 & \cdots \\ \text{doc 3} & 0 & 4 & 0 & \cdots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{array}$$

Question: How to find documents related to mathematics?

- Many (perhaps most?) relevant documents won't contain the word "mathematics"

Document modeling

Suppose you have corpus of n text documents

- Each document represented using "bag-of-words" count vectors
- $d =$ vocabulary size

$A :=$

	aardvark	abacus	abalone	$\cdots$
doc 1	3	0	0	$\cdots$
doc 2	7	0	4	$\cdots$
doc 3	0	4	0	$\cdots$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$

Question: How to find documents related to mathematics?

- Many (perhaps most?) relevant documents won't contain the word "mathematics"
- However, documents about mathematics are likely to contain word that often co-occur with "mathematics" or other math-y words (e.g., "theorem", "proof", "generality")

Latent semantic analysis

(Deerwester, Dumais, Furnas, Landauer, Harshman, 1990)

Compute rank-k SVD of A: $A_k = U_k S_k V_k^{\mathsf{T}}$

Latent semantic analysis

(Deerwester, Dumais, Furnas, Landauer, Harshman, 1990)

Compute rank-k SVD of A: $A_k = U_k S_k V_k^{\mathsf{T}}$

- Documents in corpus are represented by their corresponding rows in $U_k S_k \in \mathbb{R}^{n \times k}$

Latent semantic analysis

(Deerwester, Dumais, Furnas, Landauer, Harshman, 1990)

Compute rank-k SVD of A: $A_k = U_k S_k V_k^{\mathsf{T}}$

- Documents in corpus are represented by their corresponding rows in $U_k S_k \in \mathbb{R}^{n \times k}$
- For a new document, compute its bag-of-words vector $\vec{x} \in \mathbb{R}^d$, then compute its rank-k encoding

$$\vec{\phi}(\vec{x}) := (\vec{v}_1 \cdot \vec{x}, \ldots, \vec{v}_k \cdot \vec{x})$$

Latent semantic analysis

(Deerwester, Dumais, Furnas, Landauer, Harshman, 1990)

Compute rank-k SVD of A: $A_k = U_k S_k V_k^{\mathsf{T}}$

- Documents in corpus are represented by their corresponding rows in $U_k S_k \in \mathbb{R}^{n \times k}$
- For a new document, compute its bag-of-words vector $\vec{x} \in \mathbb{R}^d$, then compute its rank-k encoding

$$\vec{\phi}(\vec{x}) := (\vec{v}_1 \cdot \vec{x}, \ldots, \vec{v}_k \cdot \vec{x})$$

- Compare two documents $\vec{x}$ and $\vec{x}'$ by the **cosine similarity** of their encodings

$$\mathrm{sim}(\vec{x}, \vec{x}') \;=\; \frac{\vec{\phi}(\vec{x}) \cdot \vec{\phi}(\vec{x}')}{\|\vec{\phi}(\vec{x})\|_2 \|\vec{\phi}(\vec{x}')\|_2}$$

Latent semantic analysis

(Deerwester, Dumais, Furnas, Landauer, Harshman, 1990)

Compute rank-k SVD of A: $A_k = U_k S_k V_k^{\mathsf{T}}$

- Documents in corpus are represented by their corresponding rows in $U_k S_k \in \mathbb{R}^{n \times k}$
- For a new document, compute its bag-of-words vector $\vec{x} \in \mathbb{R}^d$, then compute its rank-k encoding

$$\vec{\phi}(\vec{x}) := (\vec{v}_1 \cdot \vec{x}, \ldots, \vec{v}_k \cdot \vec{x})$$

- Compare two documents $\vec{x}$ and $\vec{x}'$ by the **cosine similarity** of their encodings

$$\mathrm{sim}(\vec{x}, \vec{x}') \;=\; \frac{\vec{\phi}(\vec{x}) \cdot \vec{\phi}(\vec{x}')}{\|\vec{\phi}(\vec{x})\|_2 \|\vec{\phi}(\vec{x}')\|_2}$$

- Search queries: Construct "pseudo-document" — i.e., bag-of-words with non-zero counts for query terms — then find documents in corpus of high cosine similarity

Why low-rank approximation may be helpful

Topic model: Suppose there are k "topics" that documents might be about

Why low-rank approximation may be helpful

Topic model: Suppose there are k "topics" that documents might be about

- Each topic t specifies a distribution over words

$$\vec{c}_t = (c_{t,1}, \ldots, c_{t,d}), \qquad c_{t,j} = \Pr[\text{word } j \mid \text{topic } t]$$

Why low-rank approximation may be helpful

Topic model: Suppose there are k "topics" that documents might be about

- Each topic t specifies a distribution over words

$$\vec{c}_t = (c_{t,1}, \ldots, c_{t,d}), \qquad c_{t,j} = \Pr[\text{word } j \mid \text{topic } t]$$

- Each document i specifies a non-negative weighting over topics

$$\vec{b}_i = (b_{i,1}, \ldots, b_{i,k}), \qquad \sum_{t=1}^{k} b_{i,t} = \text{length of document } i$$

Why low-rank approximation may be helpful

Topic model: Suppose there are k "topics" that documents might be about

- Each topic t specifies a distribution over words

$$\vec{c}_t = (c_{t,1}, \ldots, c_{t,d}), \qquad c_{t,j} = \Pr[\text{word } j \mid \text{topic } t]$$

- Each document i specifies a non-negative weighting over topics

$$\vec{b}_i = (b_{i,1}, \ldots, b_{i,k}), \qquad \sum_{t=1}^{k} b_{i,t} = \text{length of document } i$$

- Document i: $b_{i,1}$ words drawn IID from topic 1, $b_{i,2}$ words drawn IID from topic 2, etc. In this model,

$$\mathbb{E}[A] = BC,$$

where

$$B = \begin{bmatrix} \leftarrow & \vec{b}_1^{\mathsf{T}} & \rightarrow \\ & \vdots & \\ \leftarrow & \vec{b}_n^{\mathsf{T}} & \rightarrow \end{bmatrix} \in \mathbb{R}^{n \times k}, \qquad C = \begin{bmatrix} \leftarrow & \vec{c}_1^{\mathsf{T}} & \rightarrow \\ & \vdots & \\ \leftarrow & \vec{c}_k^{\mathsf{T}} & \rightarrow \end{bmatrix} \in \mathbb{R}^{k \times d}$$

Why low-rank approximation may be helpful

Topic model: Suppose there are k "topics" that documents might be about

- Each topic t specifies a distribution over words

$$\vec{c}_t = (c_{t,1}, \ldots, c_{t,d}), \qquad c_{t,j} = \Pr[\text{word } j \mid \text{topic } t]$$

- Each document i specifies a non-negative weighting over topics

$$\vec{b}_i = (b_{i,1}, \ldots, b_{i,k}), \qquad \sum_{t=1}^{k} b_{i,t} = \text{length of document } i$$

- Document i: $b_{i,1}$ words drawn IID from topic 1, $b_{i,2}$ words drawn IID from topic 2, etc. In this model,

$$\mathbb{E}[A] = BC,$$

where

$$B = \begin{bmatrix} \leftarrow & \vec{b}_1^{\mathsf{T}} & \rightarrow \\ & \vdots & \\ \leftarrow & \vec{b}_n^{\mathsf{T}} & \rightarrow \end{bmatrix} \in \mathbb{R}^{n \times k}, \qquad C = \begin{bmatrix} \leftarrow & \vec{c}_1^{\mathsf{T}} & \rightarrow \\ & \vdots & \\ \leftarrow & \vec{c}_k^{\mathsf{T}} & \rightarrow \end{bmatrix} \in \mathbb{R}^{k \times d}$$

- Hence $A = BC +$ "noise"; hope that A_k avoids the noise

Caveats

- Might hope that splitting apart $A_k = U_k S_k V_k^\mathsf{T}$ as $U_k S_k$ and V_k^T will somehow give something like B and C from the topic model

  I.e., hope that $B \approx U_k S_k$ and $C \approx V_k^\mathsf{T}$

Caveats

- Might hope that splitting apart $A_k = U_k S_k V_k^\mathsf{T}$ as $U_k S_k$ and V_k^T will somehow give something like B and C from the topic model

  I.e., hope that $B \approx U_k S_k$ and $C \approx V_k^\mathsf{T}$

  - But these hopes are unlikely to be fulfilled ☹

Caveats

- Might hope that splitting apart $A_k = U_k S_k V_k^\mathsf{T}$ as $U_k S_k$ and V_k^T will somehow give something like B and C from the topic model

  I.e., hope that $B \approx U_k S_k$ and $C \approx V_k^\mathsf{T}$

  - But these hopes are unlikely to be fulfilled ☹
- Other approaches to finding low-rank approximations with "interpretable" factors:
  - Non-negative Matrix Factorization (Lee & Seung, 1999)
  - Latent Dirichlet Allocation (Pritchard, Stephens, & Donnelly, 2000; Blei, Ng, & Jordan, 2003)
  - etc.

Application: Matrix Completion

Netflix problem

Suppose ratings of movies given by users are arranged in matrix $A \in \mathbb{R}^{n \times d}$, where $A_{i,j}$ is rating given by user i for movie j

Netflix problem

Suppose ratings of movies given by users are arranged in matrix $A \in \mathbb{R}^{n \times d}$, where $A_{i,j}$ is rating given by user i for movie j

- Netflix: $n = 480000$, $d = 18000$; on average, each user rates 200 movies
- Most entries of A are "missing" (i.e., not observed)
- Goal: Predict values of missing entries

Netflix problem

Suppose ratings of movies given by users are arranged in matrix $A \in \mathbb{R}^{n \times d}$, where $A_{i,j}$ is rating given by user i for movie j

- Netflix: $n = 480000$, $d = 18000$; on average, each user rates 200 movies
- Most entries of A are "missing" (i.e., not observed)
- Goal: Predict values of missing entries

Idea: Approximate A with low-rank matrix, i.e., find

$$B = \begin{bmatrix} \leftarrow & \vec{b}_1^{\mathsf{T}} & \rightarrow \\ & \vdots & \\ \leftarrow & \vec{b}_n^{\mathsf{T}} & \rightarrow \end{bmatrix} \in \mathbb{R}^{n \times k}, \qquad C = \begin{bmatrix} \uparrow & & \uparrow \\ \vec{c}_1 & \cdots & \vec{c}_d \\ \downarrow & & \downarrow \end{bmatrix} \in \mathbb{R}^{k \times d}$$

with goal of minimizing $J(B, C) = \|A - BC\|_{\mathrm{F}}^2$

- B: "user features" (one row per user)

Netflix problem

Suppose ratings of movies given by users are arranged in matrix $A \in \mathbb{R}^{n \times d}$, where $A_{i,j}$ is rating given by user i for movie j

- Netflix: $n = 480000$, $d = 18000$; on average, each user rates 200 movies
- Most entries of A are "missing" (i.e., not observed)
- Goal: Predict values of missing entries

Idea: Approximate A with low-rank matrix, i.e., find

$$B = \begin{bmatrix} \leftarrow & \vec{b}_1^{\mathsf{T}} & \rightarrow \\ & \vdots & \\ \leftarrow & \vec{b}_n^{\mathsf{T}} & \rightarrow \end{bmatrix} \in \mathbb{R}^{n \times k}, \qquad C = \begin{bmatrix} \uparrow & & \uparrow \\ \vec{c}_1 & \cdots & \vec{c}_d \\ \downarrow & & \downarrow \end{bmatrix} \in \mathbb{R}^{k \times d}$$

with goal of minimizing $J(B, C) = \|A - BC\|_{\mathrm{F}}^2$

- B: "user features" (one row per user)
- C: "movie features" (one column per movie)

Netflix problem

Suppose ratings of movies given by users are arranged in matrix $A \in \mathbb{R}^{n \times d}$, where $A_{i,j}$ is rating given by user i for movie j

- Netflix: $n = 480000$, $d = 18000$; on average, each user rates 200 movies
- Most entries of A are "missing" (i.e., not observed)
- Goal: Predict values of missing entries

Idea: Approximate A with low-rank matrix, i.e., find

$$B = \begin{bmatrix} \leftarrow & \vec{b}_1^{\mathsf{T}} & \rightarrow \\ & \vdots & \\ \leftarrow & \vec{b}_n^{\mathsf{T}} & \rightarrow \end{bmatrix} \in \mathbb{R}^{n \times k}, \qquad C = \begin{bmatrix} \uparrow & & \uparrow \\ \vec{c}_1 & \cdots & \vec{c}_d \\ \downarrow & & \downarrow \end{bmatrix} \in \mathbb{R}^{k \times d}$$

with goal of minimizing $J(B, C) = \|A - BC\|_{\mathrm{F}}^2$

- B: "user features" (one row per user)
- C: "movie features" (one column per movie)
- Prediction of user i's rating of movie j: $\vec{b}_i \cdot \vec{c}_j$

Netflix problem

Suppose ratings of movies given by users are arranged in matrix $A \in \mathbb{R}^{n \times d}$, where $A_{i,j}$ is rating given by user i for movie j

- Netflix: $n = 480000$, $d = 18000$; on average, each user rates 200 movies
- Most entries of A are "missing" (i.e., not observed)
- Goal: Predict values of missing entries

Idea: Approximate A with low-rank matrix, i.e., find

$$B = \begin{bmatrix} \leftarrow & \vec{b}_1^{\mathsf{T}} & \rightarrow \\ & \vdots & \\ \leftarrow & \vec{b}_n^{\mathsf{T}} & \rightarrow \end{bmatrix} \in \mathbb{R}^{n \times k}, \qquad C = \begin{bmatrix} \uparrow & & \uparrow \\ \vec{c}_1 & \cdots & \vec{c}_d \\ \downarrow & & \downarrow \end{bmatrix} \in \mathbb{R}^{k \times d}$$

with goal of minimizing $J(B, C) = \|A - BC\|_{\mathrm{F}}^2$

- B: "user features" (one row per user)
- C: "movie features" (one column per movie)
- Prediction of user i's rating of movie j: $\vec{b}_i \cdot \vec{c}_j$
- Note: If all entries of A were observed, we could do this with truncated SVD

Matrix completion

Problem: We only have access to some subset $\Omega \subset [n] \times [d]$ of entries of A

Matrix completion

Problem: We only have access to some subset $\Omega \subset [n] \times [d]$ of entries of A

Common solution (used by Simon Funk in Netflix challenge): Just define objective over entries in Ω

$$J_\Omega(B, C) = \|A - BC\|_\Omega^2$$

where

$$\|M\|_\Omega := \sqrt{\sum_{(i,j)\in\Omega} M_{i,j}^2}$$

Matrix completion

Problem: We only have access to some subset $\Omega \subset [n] \times [d]$ of entries of A

Common solution (used by Simon Funk in Netflix challenge): Just define objective over entries in Ω

$$J_\Omega(B, C) = \|A - BC\|_\Omega^2$$

where

$$\|M\|_\Omega := \sqrt{\sum_{(i,j)\in\Omega} M_{i,j}^2}$$

- Now (attempt to) minimize J_Ω using gradient descent or stochastic gradient descent

Feature representations from matrix completion

- MovieLens data set ($n = 6040$ users, $d = 3952$ movies, $|\Omega| = 800000$ ratings)
- Attempted to minimize $J_\Omega(B, C)$ using stochastic gradient descent

Feature representations from matrix completion

- MovieLens data set ($n = 6040$ users, $d = 3952$ movies, $|\Omega| = 800000$ ratings)
- Attempted to minimize $J_\Omega(B, C)$ using stochastic gradient descent

Question: Are $\vec{c}_1, \ldots, \vec{c}_d \in \mathbb{R}^k$ useful feature vectors for movies?

Feature representations from matrix completion

- MovieLens data set ($n = 6040$ users, $d = 3952$ movies, $|\Omega| = 800000$ ratings)
- Attempted to minimize $J_\Omega(B, C)$ using stochastic gradient descent

Question: Are $\vec{c}_1, \ldots, \vec{c}_d \in \mathbb{R}^k$ useful feature vectors for movies?

Some close pairs of movie feature vectors $(\vec{c}_i, \vec{c}_j)$:

- Toy Story (1995), Toy Story 2 (1999)
- Sense and Sensibility (1995), Emma (1996)
- Heat (1995), Carlito's Way (1993)
- The Crow (1994), Blade (1998)
- Forrest Gump (1994), Dances with Wolves (1990)
- Mrs. Doubtfire (1993), The Bodyguard (1992)

Summary

- Many applications of low-rank approximations
- Often PCA/SVD is a good approach!
- Sometimes need to go beyond PCA/SVD