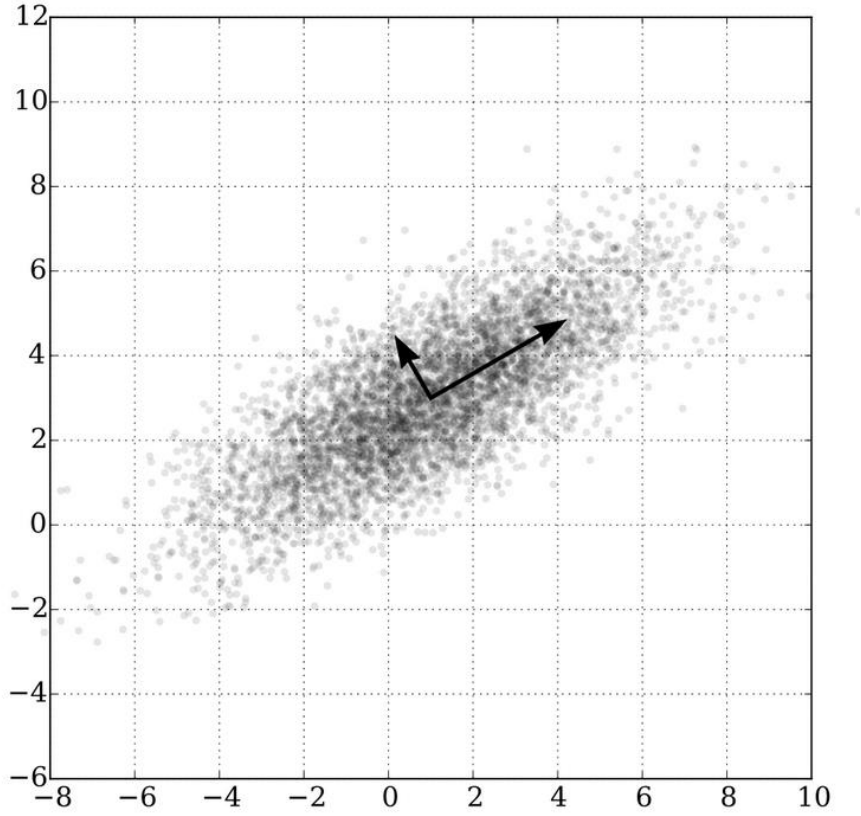
Continuous Latent Variables



Sang-wook Lee, Chae Hyeong Kim

PPT credit: Jaejoon Lee(Yonsei, ESC)

Contents

Contents

Dimension Reduction

Idea of dimensionality reduction

- 우리는 데이터를 $n \times k$행렬로 나타낸다.
- n : observation의 개수 / k : 변수의 개수
- 즉, 우리에게 차원은 변수의 개수와 같은 의미

$$\mathbf{X} = \begin{bmatrix} 1 & X_{11} & X_{12} & & X_{1,k-1} \\ 1 & X_{21} & X_{22} & \cdots & X_{2,k-1} \\ 1 & X_{31} & X_{32} & & X_{3,k-1} \\ & \vdots & & \ddots & \vdots \\ 1 & X_{n1} & X_{n2} & \cdots & X_{n,k-1} \end{bmatrix} or \begin{bmatrix} X_{11} & X_{12} & X_{13} & & X_{1k} \\ X_{21} & X_{22} & X_{23} & \cdots & X_{2k} \\ X_{31} & X_{32} & X_{33} & & X_{3k} \\ & \vdots & & \ddots & \vdots \\ X_{n1} & X_{n2} & X_{n3} & \cdots & X_{nk} \end{bmatrix}$$

$$ex) \begin{bmatrix} 1 & 0.5 & -5 & & 0.8 \\ 3 & 1 & -7 & \cdots & 1.15 \\ 2.7 & 2 & 6.2 & & 2.2 \\ & \vdots & & \ddots & \vdots \\ 4 & 5 & 9.1 & \cdots & 6 \end{bmatrix}$$

- 차원을 축소한다는 것 = 변수의 개수를 줄인다는 것

Idea of dimensionality reduction

- 목표

= k차원의 데이터를 더 작은 차원으로 나타낼 수 있을까?

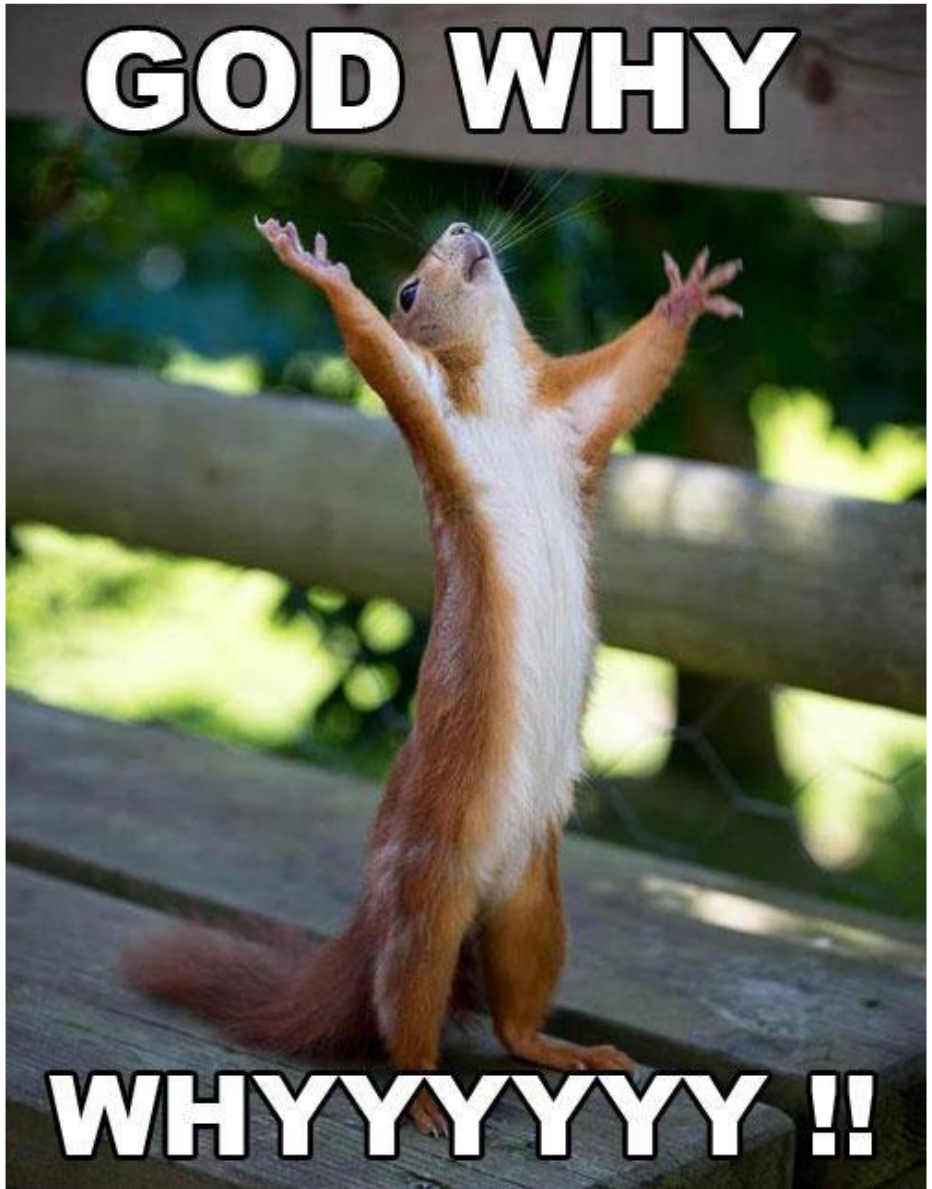
= 우리가 가진 데이터를 더 적은 개수의 변수로 요약하여 나타낼 수 있을까?

- adjusted R^2, AIC, BIC 등을 이용한 Variable Selection, 그리고 Lasso는 이러한 측면에서 변수를 선택하는 방법을 통한 차원축소로 볼 수 있다.
- 앞으로는 변수를 transform하는 방법을 통한 차원축소 방법을 살펴볼 것

⇒ Principal Component Analysis, Factor Analysis, etc.

Latent Variable을 이용한 차원축소

Why?



Taylor's Theorem

Taylor's theorem.[4][5][6] Let $k \geq 1$ be an integer and let the function $f : \mathbf{R} \to \mathbf{R}$ be k times differentiable at the point $a \in \mathbf{R}$. Then there exists a function $h_k : \mathbf{R} \to \mathbf{R}$ such that

$$f(x) = f(a) + f'(a)(x-a) + \frac{f''(a)}{2!}(x-a)^2 + \cdots + \frac{f^{(k)}(a)}{k!}(x-a)^k + h_k(x)(x-a)^k,$$

and $\lim_{x \to a} h_k(x) = 0$. This is called the **Peano form of the remainder**.

Taylor's Theorem

$$f(x) = \sum_{k=0}^{n} \frac{f^{(k)}(0)}{k!} x^k + R_n$$

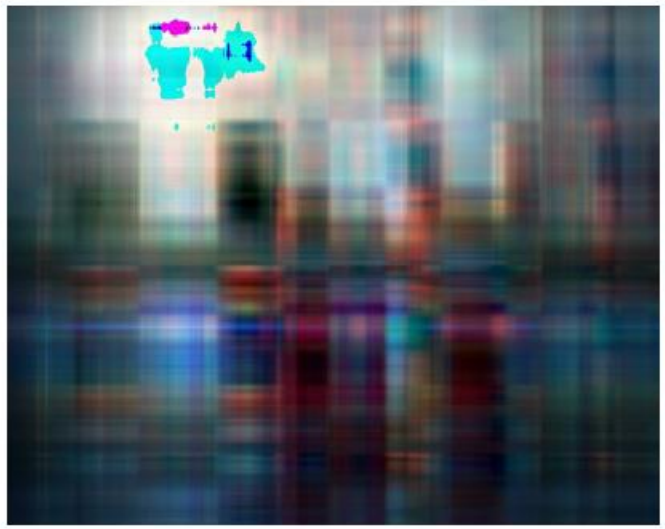
$$\rightarrow f(x) \simeq \sum_{k=0}^{n} \frac{f^{(k)}(0)}{k!} x^k = \sum_{k=0}^{n} a_k x^k$$

→ 적당한 선형결합으로 $f(x)$를 근사!

Example

avengers.jpg 1061x844 jpeg image

Thus the SVD of the image has 844 summands



rank 3

rank 45

rank 297

Contents

PCA

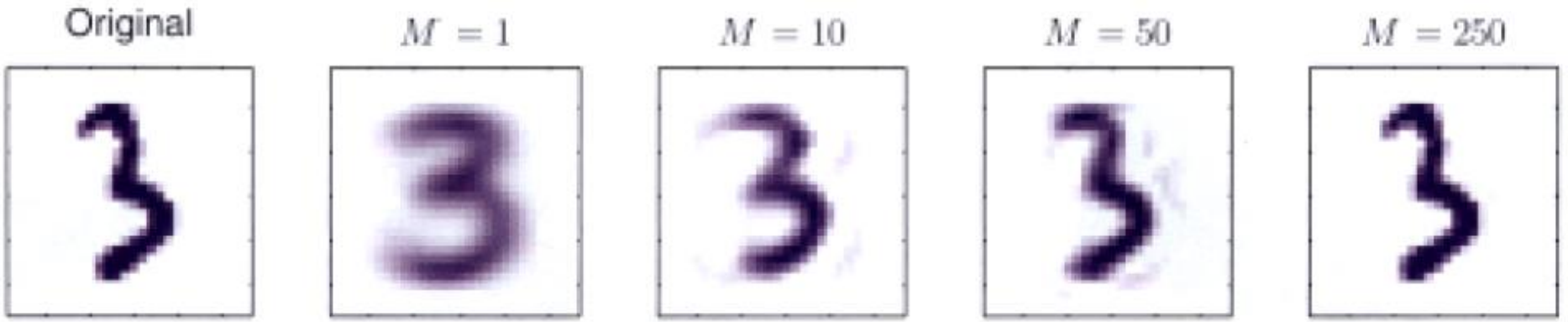


Figure 12.5 An original example from the off-line digits data set together with its PCA reconstructions obtained by retaining M principal components for various values of M. As M increases the reconstruction becomes more accurate and would become perfect when $M = D = 28 \times 28 = 784$.

Principal Component Analysis

- 목표 : 우리의 data를 잘 나타내는 Principal Component(주성분)을 찾고, 이를 통해 data를 요약하여 나타내겠다.

- feature selection과 같이 일부 변수를 고르는 것이 아니라, 기존 변수들의 선형 결합으로 이루어진 새로운 변수를 만드는 것.

Assumptions of PCA

1. Data는 서로 선형 관계에 있다

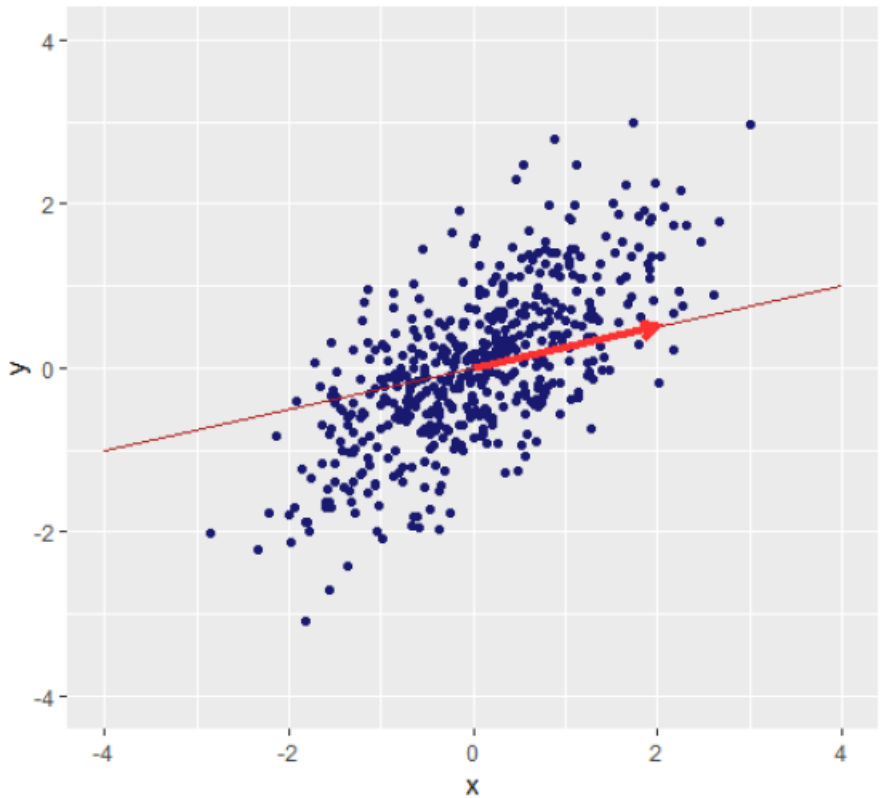
2. Projection 후 *variance가 더 클 수록* 더 큰 중요성을 갖는다

Why?

3. 데이터는 centered and scaled (단위 조정)

→ 공분산 행렬이 $S = \frac{1}{n-1} X'X$

Principal Component의 조건



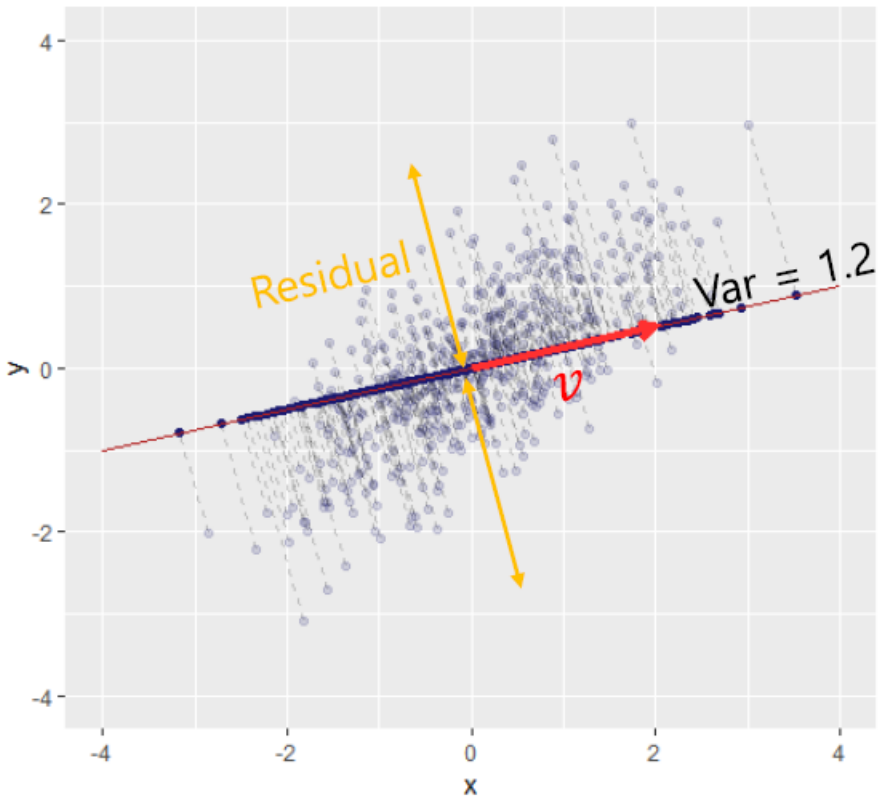
- 다음과 같은 2차원 상의 자료가 있다.

  ⇒ 2개의 변수를 갖는 자료

- 이 자료를 1차원인 선 위에 projection 하여, 즉 1개의 변수로 요약하여 나타낼 수 있을까?

- 붉은 화살표로 나타낸 임의의 벡터, $v = (2, 0.5)$ 위에 이 자료를 projection 하여 이 자료를 요약한다고 해보자.

Principal Component의 조건



- 벡터 v 위로 projection을 한 데이터요약은 자료의 변동성, 즉 분산을 1.2만큼 포착(capture)했다.

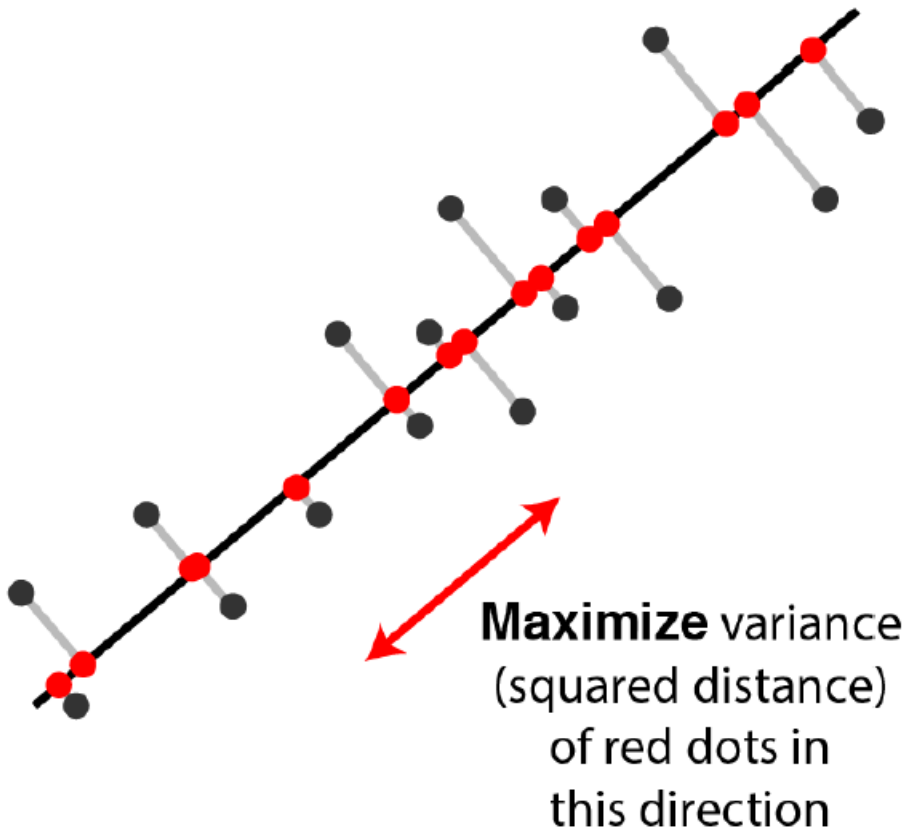
- projection을 거친 후, 자료의 변동성을 최대한 잘 포착하는 하나의 벡터 v를 찾자.

⇒ 조건 ① : projection된 데이터의 분산을 최대화하는 벡터 v를 고르자!

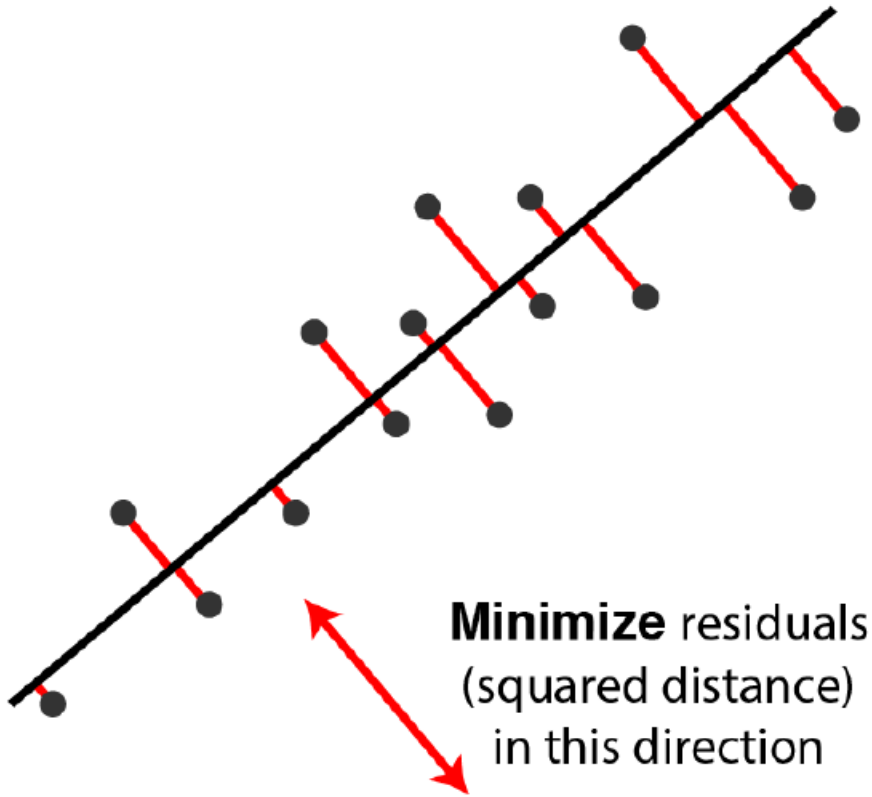
- 양의 상관관계를 포착하기는 하였으나, 요약 후에는 정보를 잃어버리게 되는 residual이 존재.

⇒ 조건 ② : projection 후의 residual을 최소화하는 벡터 v를 고르자!

① projection된 데이터의 variance를 최대화 하는 벡터 v 를 고르자!



② projection후의 residual을 최소화하는 벡터 v 를 고르자



Projection

- $\mathbb{R}^2$의 점 4개를 나타내는 data matrix $\mathbf{X}$, $\mathbb{R}^2$의 한 단위벡터 $\mathbf{c}$

$$\mathbf{X} = \begin{bmatrix} 0 & 1 \\ 0 & 2 \\ -1 & -2 \\ -3 & -1 \end{bmatrix} \qquad \mathbf{c} = \begin{bmatrix} 1/\sqrt{2} \\ 1/\sqrt{2} \end{bmatrix}$$

- 행렬 $\mathbf{X}$의 점을 벡터 $\mathbf{c}$로 생성된 직선 위에 projection한다면?

$$\mathbf{Xc} = \begin{bmatrix} 0 & 1 \\ 0 & 2 \\ -1 & -2 \\ -3 & -1 \end{bmatrix} \begin{bmatrix} 1/\sqrt{2} \\ 1/\sqrt{2} \end{bmatrix} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 2 \\ -3 \\ -4 \end{bmatrix}$$

: 이 4개의 값은 새로운 축 $\mathbf{c}$ 위에서의 좌표를 의미

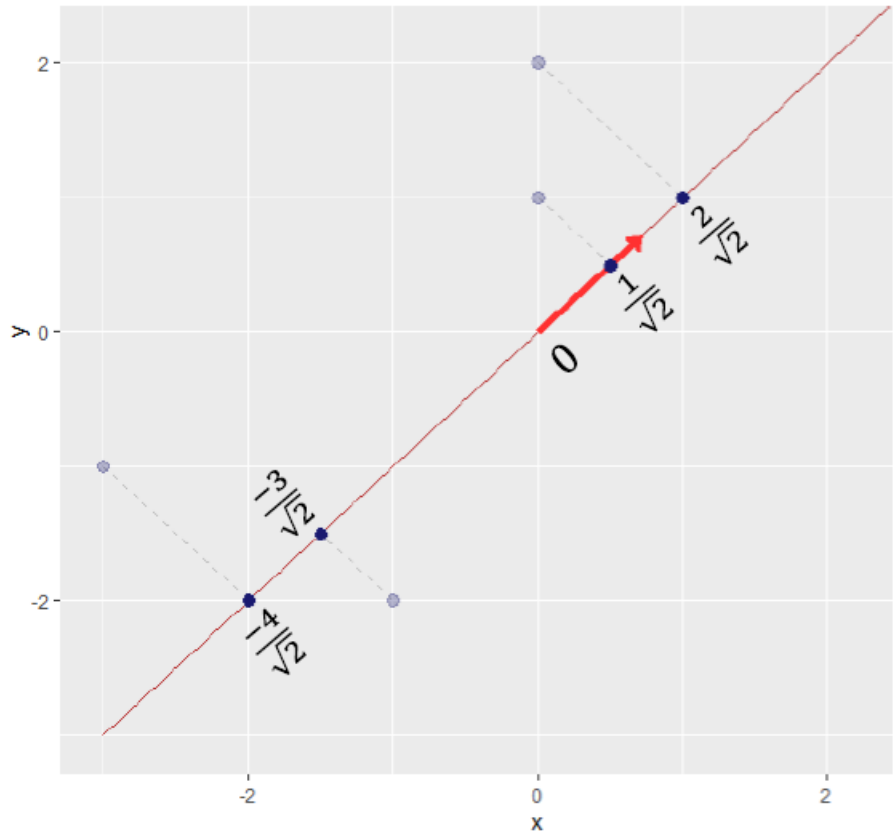
Why?

Projection

$$\mathbf{X} = \begin{bmatrix} 0 & 1 \\ 0 & 2 \\ -1 & -2 \\ -3 & -1 \end{bmatrix} \qquad \mathbf{c} = \begin{bmatrix} 1/\sqrt{2} \\ 1/\sqrt{2} \end{bmatrix}$$

$$\mathbf{Xc} = \begin{bmatrix} 0 & 1 \\ 0 & 2 \\ -1 & -2 \\ -3 & -1 \end{bmatrix} \begin{bmatrix} 1/\sqrt{2} \\ 1/\sqrt{2} \end{bmatrix} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 2 \\ -3 \\ -4 \end{bmatrix}$$

: 새로운 축 **c** 위에서의 좌표



Projection

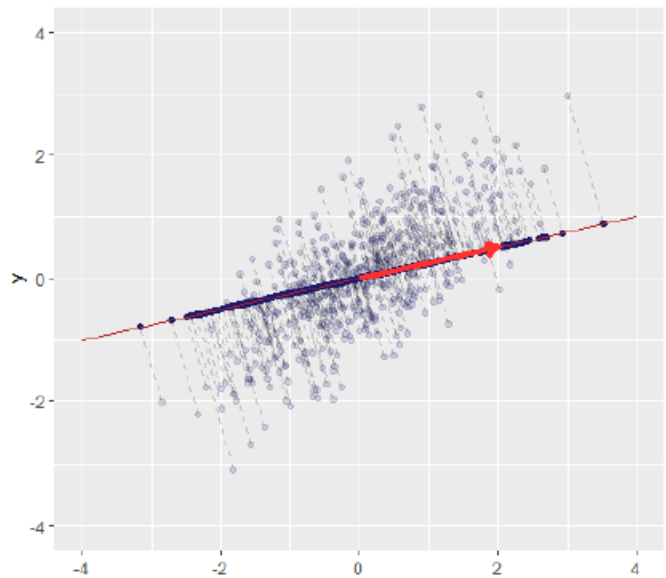
- $\mathbb{R}^k$의 점 n개를 나타내는 $(n \times k)$ data matrix $\mathbf{X}$, $\mathbb{R}^k$의 한 단위벡터 $\mathbf{c}$
- 행렬 $\mathbf{X}$의 점들을 벡터 $\mathbf{c}$로 생성된 직선 위에 projection한다면?

⇒ 새로운 축 위에서의 좌표는 $(n \times 1)$ 벡터 $\mathbf{Xc}$ 로 나타낼 수 있다.

- projection 후의 데이터, $\mathbf{Xc}$의 분산은 다음과 같다.

$$\mathrm{Var}(\mathbf{Xc}) = (\mathbf{Xc})^{\mathrm{T}}(\mathbf{Xc}) = \mathbf{c}^{\mathrm{T}}\left(\mathbf{X}^{\mathrm{T}}\mathbf{X}\right)\mathbf{c}$$

- 우리는 projection 후의 데이터의 분산이 최대가 되는 벡터 $\mathbf{c}$를 찾기로 했다.



조건 ① projection된 데이터의 분산을 최대화하는 벡터 c 를 고르자!

$$\underset{\mathbf{c}}{maximize} \quad \mathrm{Var}(\mathbf{Xc}) = \mathbf{c}^{\mathrm{T}}\left(\mathbf{X}^{\mathrm{T}}\mathbf{X}\right)\mathbf{c} \quad \text{subject to} \quad \mathbf{c}^{\mathrm{T}}\mathbf{c} = 1$$

- 여기서 $\mathbf{X}$의 (표본)공분산행렬을 $\boldsymbol{S}$라고 할 때, $\boldsymbol{S} = \frac{1}{n-1}\mathbf{X}^{\mathrm{T}}\mathbf{X}$ 이므로

$$\Leftrightarrow \quad \underset{\mathbf{c}}{maximize} \quad \mathrm{Var}(\mathbf{Xc}) = \mathbf{c}^{\mathrm{T}}\mathbf{Sc} \quad \text{subject to} \quad \mathbf{c}^{\mathrm{T}}\mathbf{c} = 1$$

- 라그랑지 승수법을 이용하여 해를 구하면 다음과 같다. 결과만 안다면 자세한 증명은 넘어가도 괜찮다

Let $\mathcal{L} = \mathbf{c}^{\mathrm{T}}\boldsymbol{S}\mathbf{c} - \lambda\left(\mathbf{c}^{\mathrm{T}}\mathbf{c} - 1\right)$, $\quad \frac{\partial \mathcal{L}}{\partial \mathbf{c}} = 2\boldsymbol{S}\mathbf{c} - 2\lambda\mathbf{c} = 0$ 즉, $\boldsymbol{S}\mathbf{c} = \lambda\mathbf{c}$일 때 $\mathrm{Var}(\mathbf{Xc})$ 최대

이를 다시 목적함수에 대입하면 $\mathrm{Var}(\mathbf{Xc}) = \mathbf{c}^{\mathrm{T}}\boldsymbol{S}\mathbf{c} = \mathbf{c}^{\mathrm{T}}(\lambda\mathbf{c}) = \lambda\mathbf{c}^{\mathrm{T}}\mathbf{c} = \lambda \cdot 1 = \lambda$

- **결론**

$\mathbf{X}$의 공분산행렬의 Eigenvector에 projection할 때, projection된 데이터의 분산 최대화
그 때의 분산 값은 해당 Eigenvector의 Eigenvalue의 값과 같다.

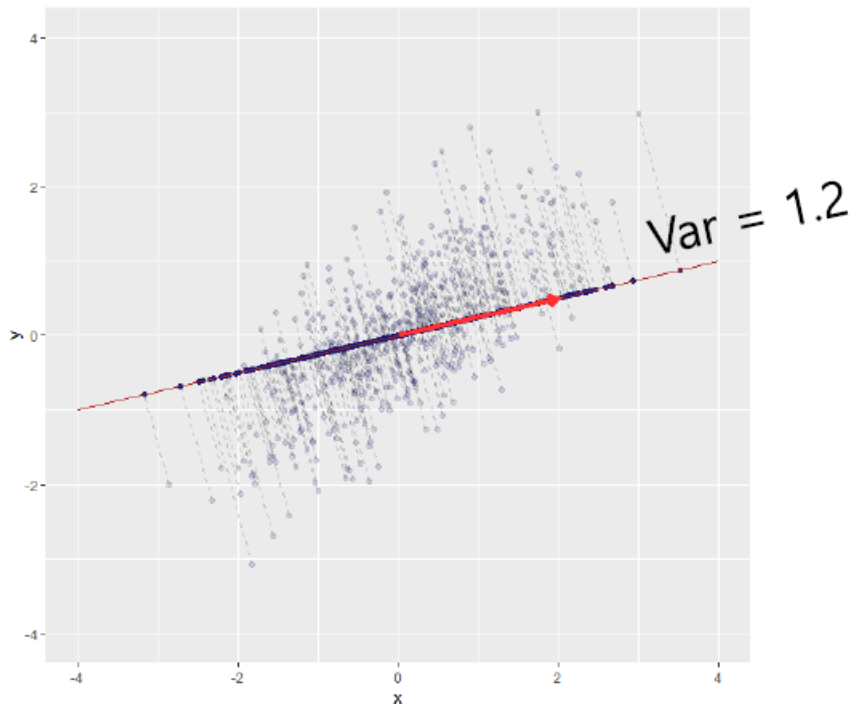
(심화학습): Matrix norm $||A||_2$

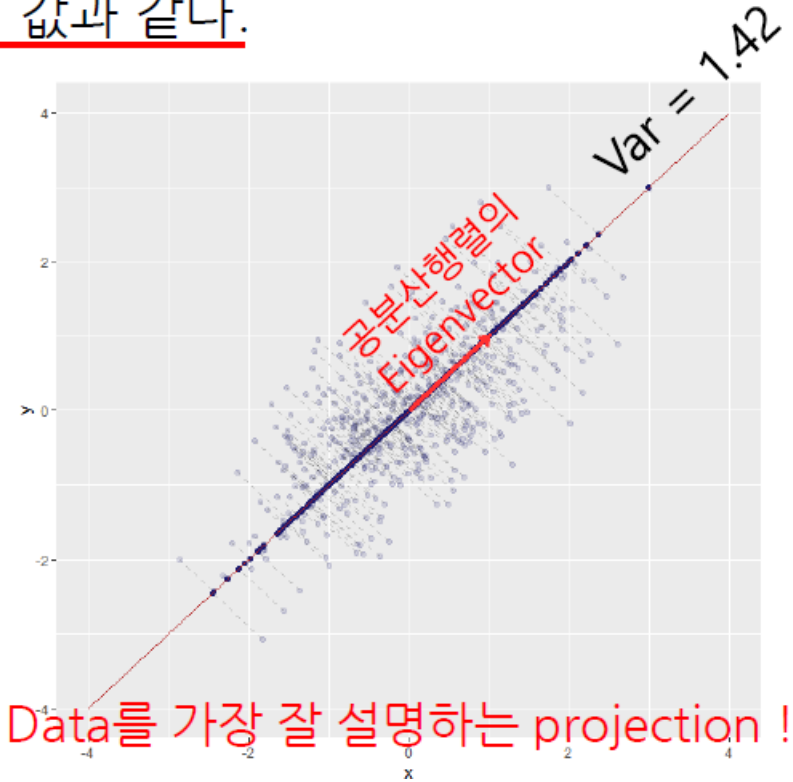
조건 ① projection된 데이터의 분산을 최대화하는 벡터 c 를 고르자!

- 결론

X의 공분산행렬의 Eigenvector에 projection할 때, projection된 데이터의 분산 최대화

그 때의 분산 값은 해당 Eigenvector의 Eigenvalue의 값과 같다.





Data를 가장 잘 설명하는 projection !

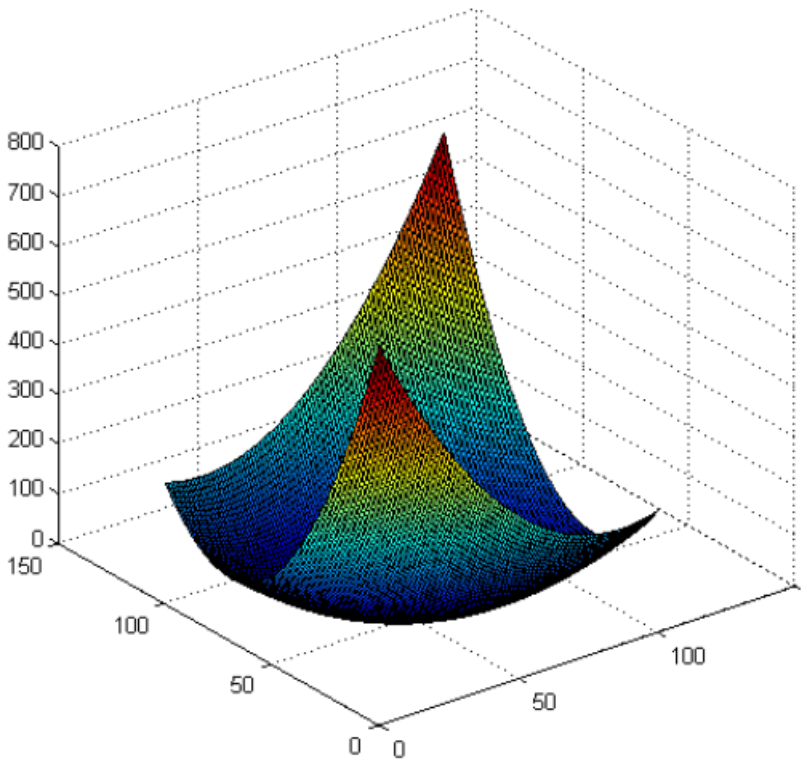
Eigenvalue and Eigenvector

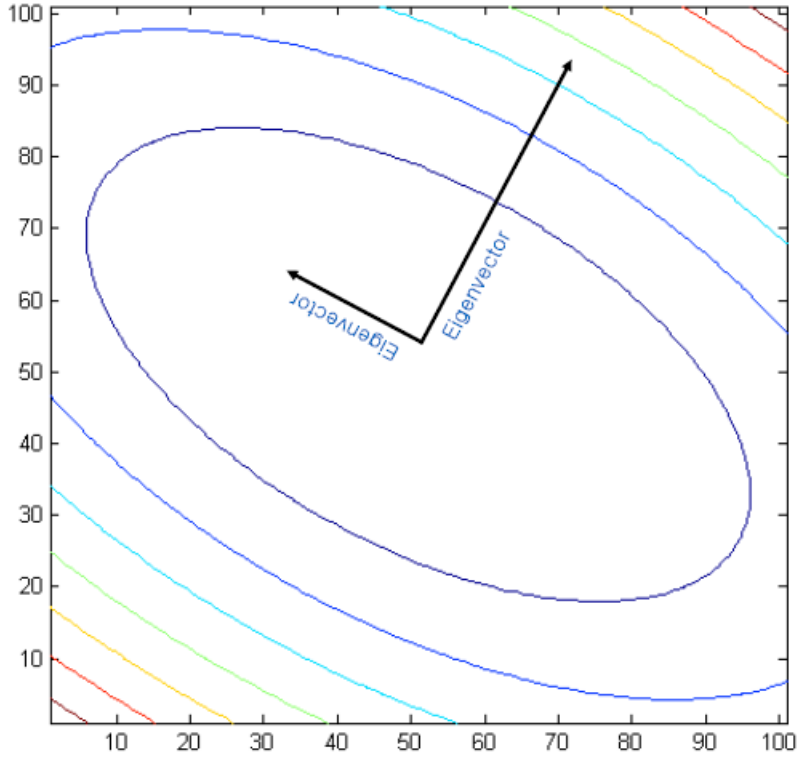
Eigenvalues

$$\mathbf{A} = \begin{bmatrix} 3.25 & 1.30 \\ 1.30 & 1.75 \end{bmatrix} = \begin{bmatrix} 0.50 & -0.87 \\ -0.87 & -0.50 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 4 \end{bmatrix} \begin{bmatrix} 0.50 & -0.87 \\ -0.87 & -0.50 \end{bmatrix}^T$$

Eigenvectors

Eigenvectors





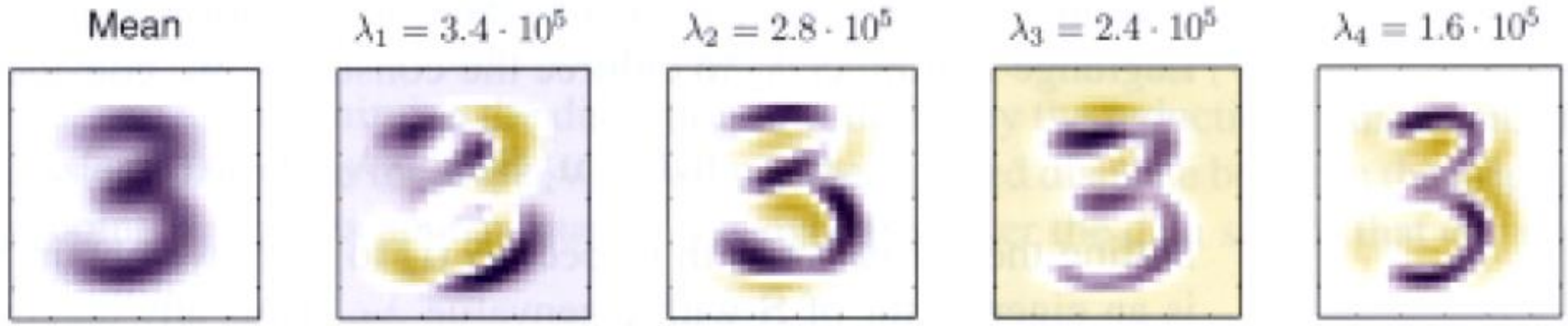
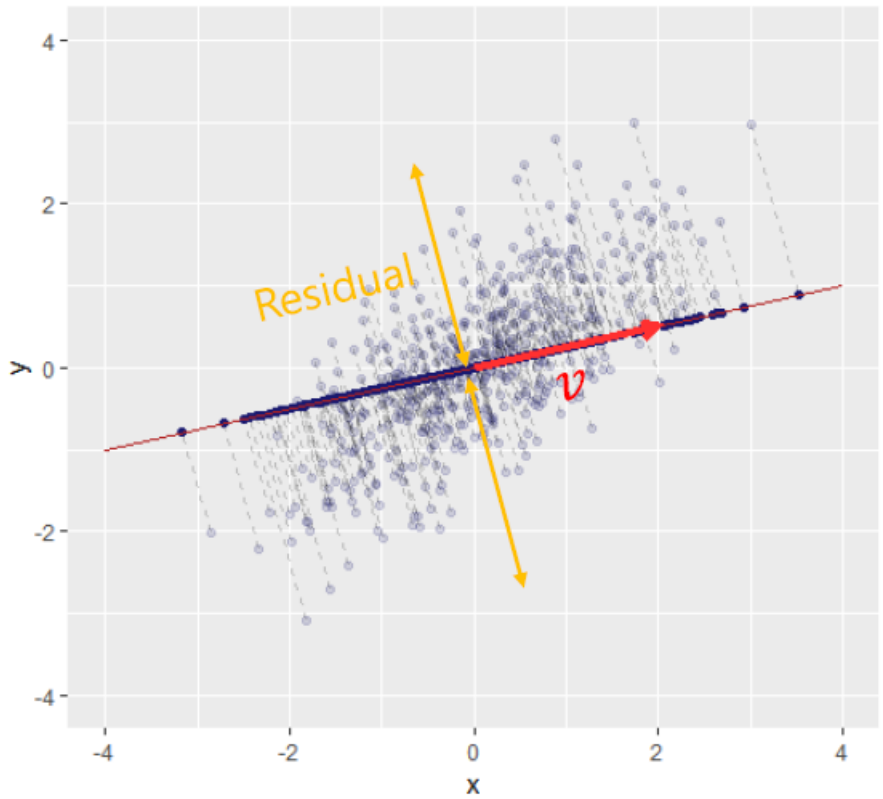


Figure 12.3 The mean vector $\overline{\mathbf{x}}$ along with the first four PCA eigenvectors $\mathbf{u}_1, \ldots, \mathbf{u}_4$ for the off-line digits data set, together with the corresponding eigenvalues.

조건 ② projection후 잃어버리는 residual을 최소화하는 벡터를 고르자!

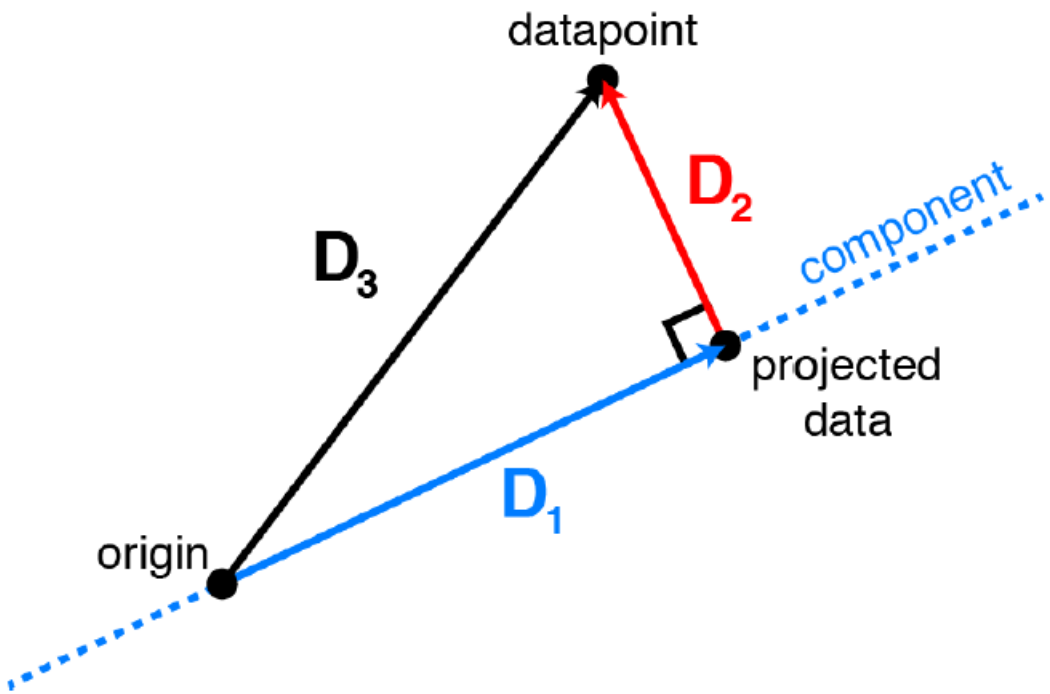


- projection을 통한 차원축소 후, residual 에 해당하는 부분의 정보는 잃어버리게 된다.
- 이 정보의 손실를 최소화하는 것 역시 Principal Component의 중요한 조건
- 다행히도 조건 ①을 만족하는 벡터는 조건 ② 역시 자동적으로 만족한다.
- 결국 두 조건은 equivalent한 조건이었던 것! **증명해보자**

조건 ① projection된 데이터의 분산을 최대화하는 벡터를 고르자!

||

조건 ② projection후 잃어버리는 residual을 최소화하는 벡터를 고르자!



$$D_3^2 = D_1^2 + D_2^2$$

$$\text{initial variance} = \text{remaining variance} + \text{lost variance}$$

$$\|a_i\|^2 = \|w_i c\|^2 + \|a_i - w_i c\|^2$$

this is constant — maximize this **or** minimize this

projection 후 포착한 분산을 최대화 ⇔ projection하면서 잃어버린 residual 최소화

Summary of PCA

1. Principal Component(PC)는 공분산 행렬 S 의 eigenvector들

2. eigenvector에 projection하여 포착된 분산값은 해당 eigenvalue

3. 따라서 eigenvalue가 가장 큰 eigenvector가 가장 좋은 PC

4. k개의 eigenvector중 eigenvalue가 가장 큰 d (d < k)개에 projection하자

How to compute PCA?

- 방법 ① : $\mathbf{X}$의 공분산행렬인 $\boldsymbol{S}$의 Eigenvector와 Eigenvalue를 직접 다 찾는다
  ⇒ 정말 느린 알고리즘. 사용 못 할 정도. bye bye

- 방법 ② : 특이값 분해(Singular Value Decomposition, SVD)
  ⇒ 빠르게 상위 d개의 공분산행렬인 $\boldsymbol{S}$의 Eigenvector를 구해낼 수 있다.
  ⇒ $\mathbf{X}$의 SVD한 결과에 $\mathbf{X}^{\mathrm{T}}\mathbf{X}$의 eigenvector들로 이루어진 행렬이 있는 점을 이용.

$$\underset{(n\times k)}{\mathbf{X}} = \begin{bmatrix} X_{11} & X_{12} & X_{13} & \cdots & X_{1k} \\ X_{21} & X_{22} & X_{23} & & X_{2k} \\ X_{31} & X_{32} & X_{33} & & X_{3k} \\ & \vdots & & \ddots & \vdots \\ X_{n1} & X_{n2} & X_{n3} & \cdots & X_{nk} \end{bmatrix} \quad \underset{(k\times d)}{\mathbf{C}} = \begin{bmatrix} \mathbf{c}_1 & \cdots & \mathbf{c}_d \end{bmatrix} \Rightarrow \underset{(n\times d)}{\mathbf{Z} = \mathbf{XC}} = \begin{bmatrix} \mathbf{z}_1 & \cdots & \mathbf{z}_d \end{bmatrix}$$

그 외에 MLE, EM algorithm 등이 있다. 그 중 EM에 대해 알아보자.

EM Algorithm

$$\ln p\left(\mathbf{X}, \mathbf{Z}|\boldsymbol{\mu}, \mathbf{W}, \sigma^2\right) = \sum_{n=1}^{N} \{\ln p(\mathbf{x}_n|\mathbf{z}_n) + \ln p(\mathbf{z}_n)\} \tag{12.52}$$

where the n^{th} row of the matrix $\mathbf{Z}$ is given by $\mathbf{z}_n$. We already know that the exact maximum likelihood solution for $\boldsymbol{\mu}$ is given by the sample mean $\overline{\mathbf{x}}$ defined by (12.1), and it is convenient to substitute for $\boldsymbol{\mu}$ at this stage. Making use of the expressions (12.31) and (12.32) for the latent and conditional distributions, respectively, and taking the expectation with respect to the posterior distribution over the latent variables, we obtain

$$\begin{aligned}\mathbb{E}[\ln p\left(\mathbf{X}, \mathbf{Z}|\boldsymbol{\mu}, \mathbf{W}, \sigma^2\right)] = -\sum_{n=1}^{N} \Bigg\{ & \frac{D}{2}\ln(2\pi\sigma^2) + \frac{1}{2}\mathrm{Tr}\left(\mathbb{E}[\mathbf{z}_n\mathbf{z}_n^{\mathrm{T}}]\right) \\ & + \frac{1}{2\sigma^2}\|\mathbf{x}_n - \boldsymbol{\mu}\|^2 - \frac{1}{\sigma^2}\mathbb{E}[\mathbf{z}_n]^{\mathrm{T}}\mathbf{W}^{\mathrm{T}}(\mathbf{x}_n - \boldsymbol{\mu}) \\ & + \frac{1}{2\sigma^2}\mathrm{Tr}\left(\mathbb{E}[\mathbf{z}_n\mathbf{z}_n^{\mathrm{T}}]\mathbf{W}^{\mathrm{T}}\mathbf{W}\right)\Bigg\}. \end{aligned} \tag{12.53}$$

Note that this depends on the posterior distribution only through the sufficient statistics of the Gaussian. Thus in the E step, we use the old parameter values to evaluate

$$\mathbb{E}[\mathbf{z}_n] = \mathbf{M}^{-1}\mathbf{W}^{\mathrm{T}}(\mathbf{x}_n - \overline{\mathbf{x}}) \tag{12.54}$$

$$\mathbb{E}[\mathbf{z}_n\mathbf{z}_n^{\mathrm{T}}] = \sigma^2\mathbf{M}^{-1} + \mathbb{E}[\mathbf{z}_n]\mathbb{E}[\mathbf{z}_n]^{\mathrm{T}} \tag{12.55}$$

which follow directly from the posterior distribution (12.42) together with the standard result $\mathbb{E}[\mathbf{z}_n\mathbf{z}_n^{\mathrm{T}}] = \mathrm{cov}[\mathbf{z}_n] + \mathbb{E}[\mathbf{z}_n]\mathbb{E}[\mathbf{z}_n]^{\mathrm{T}}$. Here $\mathbf{M}$ is defined by (12.41).

In the M step, we maximize with respect to $\mathbf{W}$ and σ^2, keeping the posterior statistics fixed. Maximization with respect to σ^2 is straightforward. For the maximization with respect to $\mathbf{W}$ we make use of (C.24), and obtain the M-step equations

$$\mathbf{W}_{\text{new}} = \left[\sum_{n=1}^{N}(\mathbf{x}_n - \overline{\mathbf{x}})\mathbb{E}[\mathbf{z}_n]^{\mathrm{T}}\right]\left[\sum_{n=1}^{N}\mathbb{E}[\mathbf{z}_n\mathbf{z}_n^{\mathrm{T}}]\right]^{-1} \tag{12.56}$$

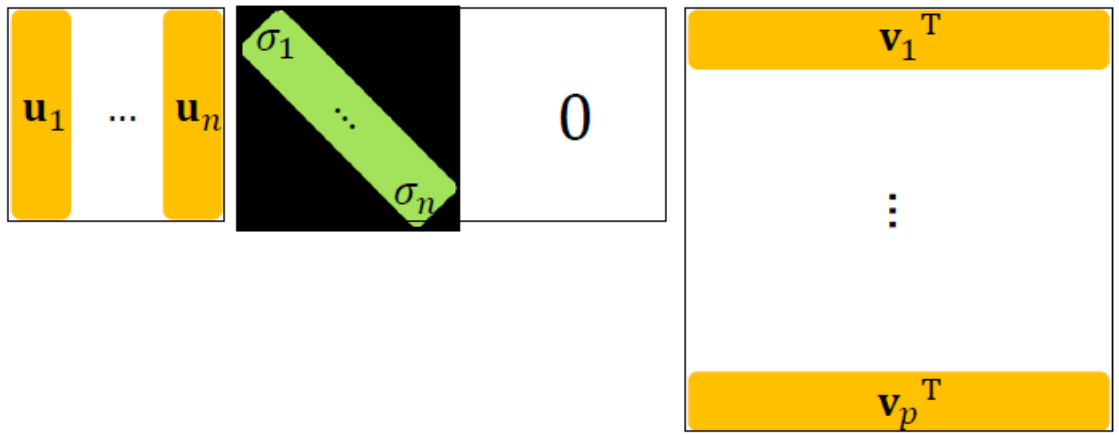
$$\begin{aligned}\sigma^2_{\text{new}} = \frac{1}{ND}\sum_{n=1}^{N}\Big\{ & \|\mathbf{x}_n - \overline{\mathbf{x}}\|^2 - 2\mathbb{E}[\mathbf{z}_n]^{\mathrm{T}}\mathbf{W}_{\text{new}}^{\mathrm{T}}(\mathbf{x}_n - \overline{\mathbf{x}}) \\ & + \mathrm{Tr}\left(\mathbb{E}[\mathbf{z}_n\mathbf{z}_n^{\mathrm{T}}]\mathbf{W}_{\text{new}}^{\mathrm{T}}\mathbf{W}_{\text{new}}\right)\Big\}. \end{aligned} \tag{12.57}$$

…그만 알아보자

SVD Revisited

변수가 자료보다 더 많은 상황($n < p$)의 그림

$$\underset{(n \times p)}{\mathbf{X}} = \underset{(n \times n)}{\mathrm{U}} \quad \underset{(n \times p)}{\Sigma} \quad \underset{(p \times p)}{\mathrm{V}^{\mathrm{T}}} =$$



U : XX^{T} 의 eigenvector들을 세로로 쌓은 행렬

$\mathbf{u}_1, \ldots, \mathbf{u}_n$: XX^{T} 의 eigenvector

V : $\mathrm{X}^{\mathrm{T}}\mathrm{X}$ 의 eigenvector들을 세로로 쌓은 행렬

$\mathbf{v}_1, \ldots, \mathbf{v}_p$: $\mathrm{X}^{\mathrm{T}}\mathrm{X}$ 의 eigenvector

Σ : $\mathrm{X}^{\mathrm{T}}\mathrm{X}$ 와 XX^{T}의 0이 아닌 eigenvalue들을 크기 내림차순으로 대각원소에 넣은 행렬

Fact. $\mathrm{X}^{\mathrm{T}}\mathrm{X}$ 와 XX^{T} 는 0이 아닌 eigenvalue들이 모두 같다. 그리고 그 공통된 eigenvalue들이 모두 양수

Truncated SVD

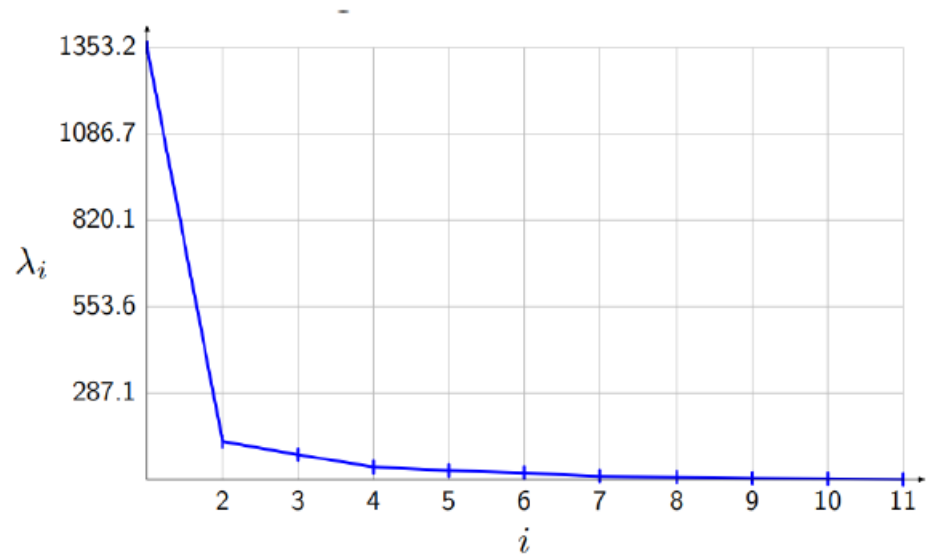
$$X_{n\times k} = \sigma_1 u_1 v_1' + \sigma_2 u_2 v_2' + \cdots + \sigma_d u_d v_d' + \cdots + \sigma_k u_k v_k'$$

$$\downarrow$$

$$X_{n\times k} \simeq \sigma_1 u_1 v_1' + \sigma_2 u_2 v_2' + \cdots + \sigma_d u_d v_d'$$

How many principal components?

- Eigenvalue의 크기는 그 Eigenvector(= 주성분)이 포착한 분산의 크기이다.
- Eigenvalue의 크기가 클 수록 좋은 것



- 주로 Eigenvalue는 그 크기가 급감
- 즉, Data matrix $\mathbf{X}$ 를 요약하는데 그렇게 많은 주성분이 필요하지 않다.

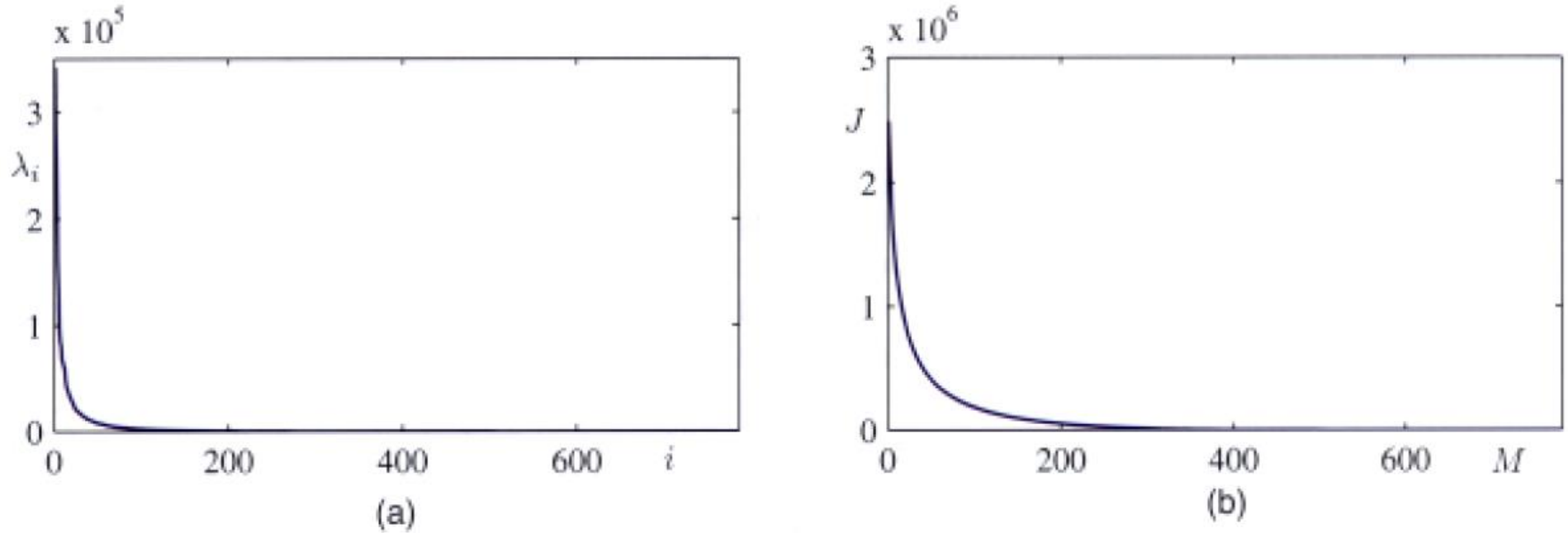


Figure 12.4 (a) Plot of the eigenvalue spectrum for the off-line digits data set. (b) Plot of the sum of the discarded eigenvalues, which represents the sum-of-squares distortion J introduced by projecting the data onto a principal component subspace of dimensionality M.

Example

Seolgwangeun.jpg 750x750 jpeg image

Thus the SVD of the image has 750 summands

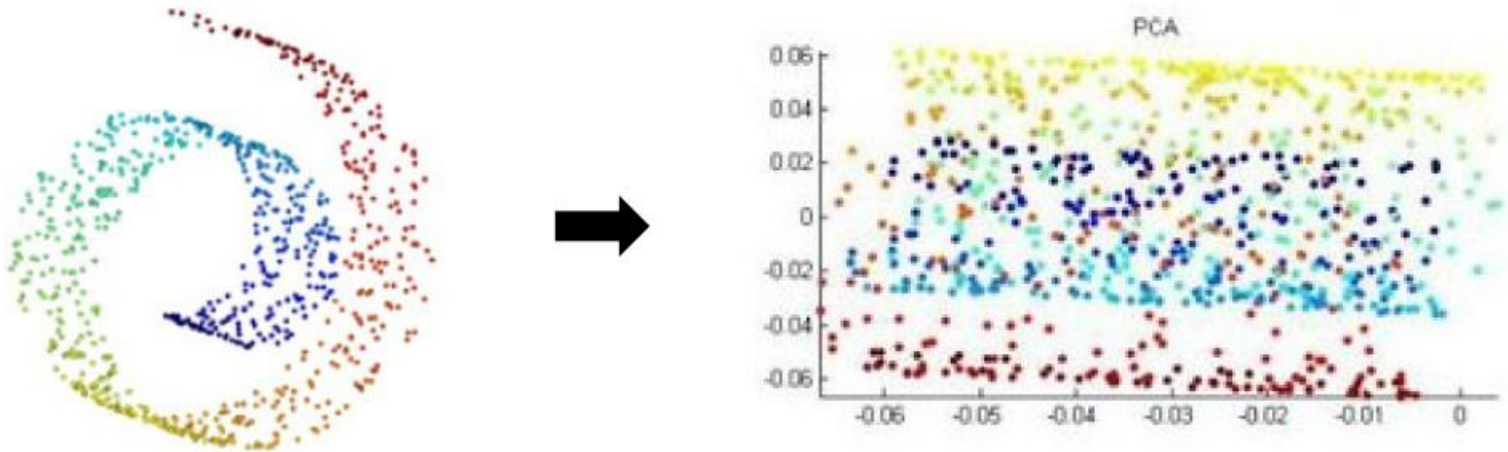
rank 3

rank 45

rank 297

Limitations

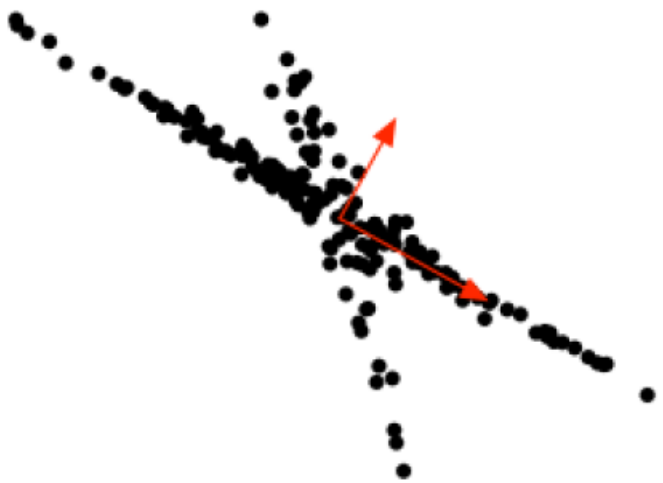
1. PCA는 Principal component들이 기존 설명변수들의 선형결합이므로, 다음과 같은 예시의 비선형 관계의 data에서는 PCA를 이용한 분석의 결과가 유의미하지 않을 수 있음.



이 경우 non-linear PCA 등의 방법이 있음

Limitations

2. PCA는 공분산행렬의 eigenvector이기 때문에, PCA를 수행하여 얻은 principal component는 항상 서로 orthogonal하다.



빨강색 벡터 : Principal components

검은색 점으로 나타난 data를 orthogonal한 Principal component들로는 잘 나타낼 수 없는 경우도 있다.

Limitations

3. 분석 결과 얻어진 Principal component들을 어떻게 해석할 것인가? 분석 결과의 계수들을 보고 principal component의 의미를 해석하기 어려운 경우가 있다는 문제가 있다.

ex) 다음 principal component를 어떻게 해석할 것인가?

$$PC_1 = -0.044 \cdot X_1 + 0.245 \cdot X_2 + 0.002 \cdot X_3 + 0.239 \cdot X_4 - 0.142 \cdot X_5 - 0.395 \cdot X_6$$

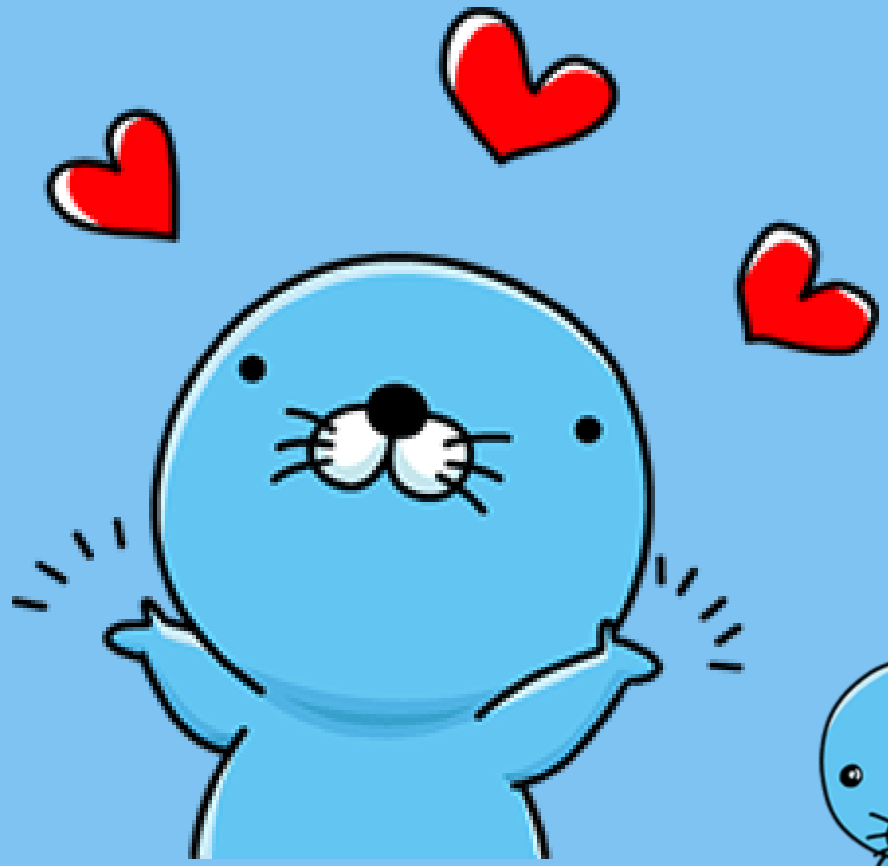
이 해석을 더 용이하게 하기 위해, PCA에 lasso penalty를 접목하여 좀더 sparse한 계수의 분석 결과를 얻고자 하는 sparse PCA라는 모형도 있음.

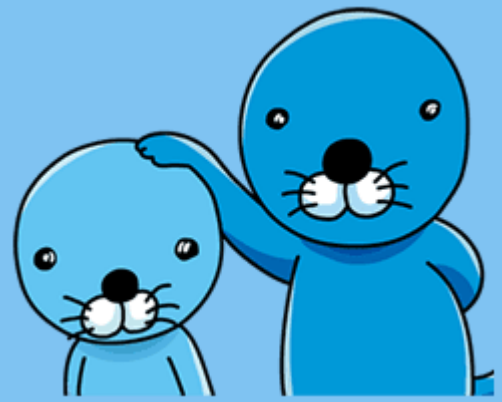
분석 결과에 0이 많은,
위의 경우는 빨간 표시 된 계수가 0이 될 것

prediction이 목표라면 큰 문제는 아니지만, 통계나 경제처럼 해석, modeling이 중요한 도메인에서는 큰 문제

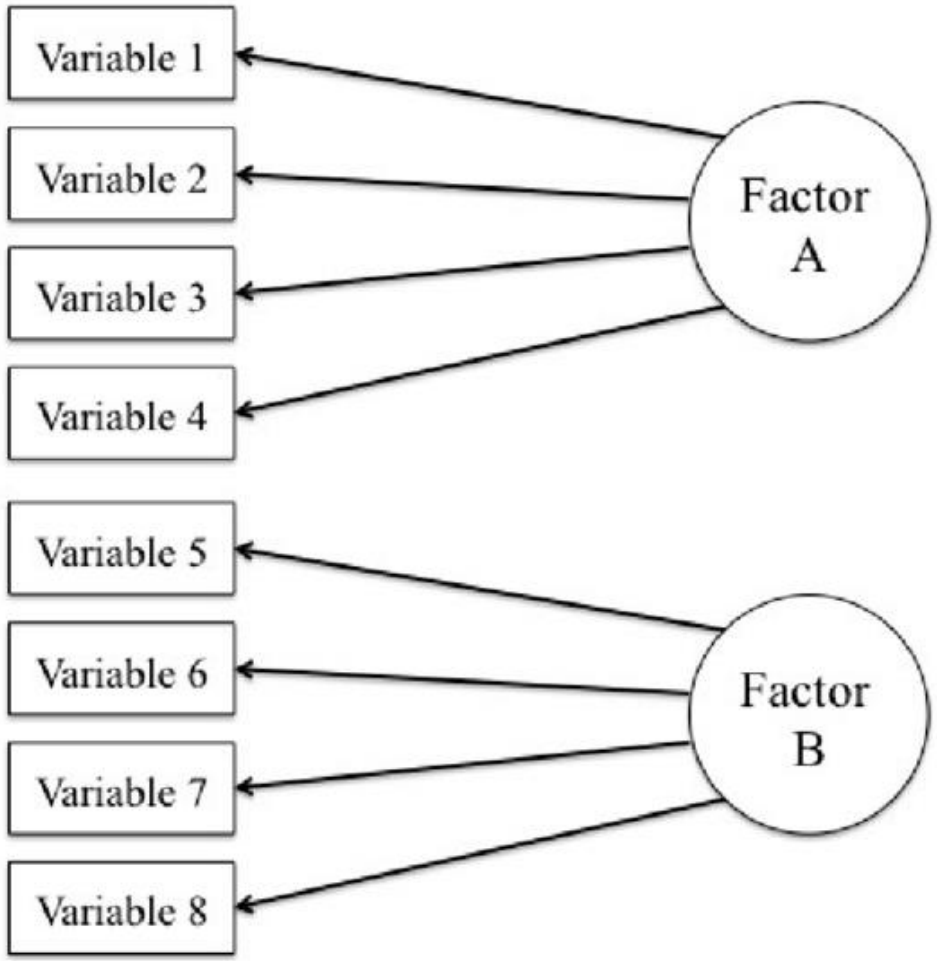
Contents

쉬었다 합시다 ㅎㅎ





FA


Variable 1
Variable 2
Variable 3
Variable 4
Variable 5
Variable 6
Variable 7
Variable 8
Factor A
Factor B

Compared to PCA

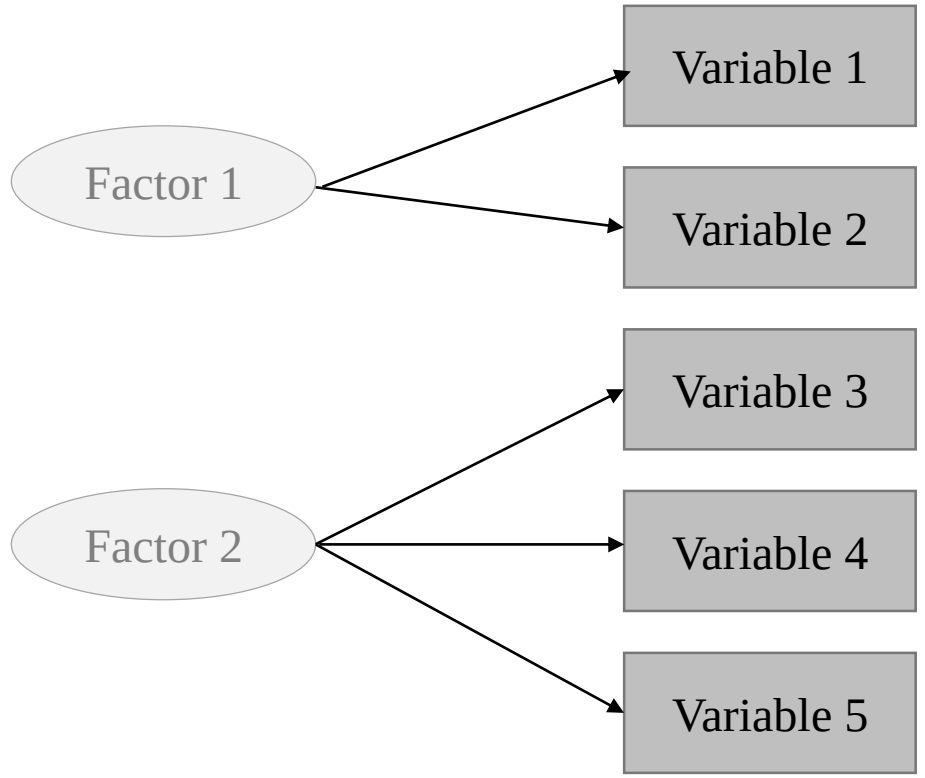
- PCA는 Data의 분산을 가장 잘 반영하는 벡터 d개($d < K$)를 찾아 projection 후 선형결합. (분산 중심의 분석). 반면 FA는 공분산 중심의 분석

- 잠재변수(Latent Variable): 직접적으로 관측은 할 수 없지만, 목적변수에 영향을 주는 것으로 여겨지는 변수. ex) 삶의 질, 행복

- FA는 잠재변수에 의해 X변수들이 결정된다고 가정한다.

FA

$$\begin{pmatrix} 1.00 & .90 & .05 & .05 & .05 \\ .90 & 1.00 & .05 & .05 & .05 \\ .05 & .05 & 1.00 & .90 & .90 \\ .05 & .05 & .90 & 1.00 & .90 \\ .05 & .05 & .90 & .90 & 1.00 \end{pmatrix}$$



Unobserved!

Observed!

ex) 국어, 미적분, 확률통계, 영어, 제2외국어 성적에 대한 데이터가 있을 때,

이 다섯 개 과목 변수가 사실 어학능력과 수리능력이라는 두 개의 Factor에 의해 결정된다면?

Factor analysis : Model

- mean이 $\mu_1, \mu_2, \ldots, \mu_p$인 p개의 관측가능한 예측변수, $X_1, X_2, \ldots, X_p$,
- d개의 관측불가능한 잠재변수(latent), $F_1, F_2, \ldots, F_d$,가 있다. 이 잠재변수들을 요인분석에서는 "common factors"라고 부른다.
- 예측변수들은 "Factor들의 선형결합 + 오차항"의 꼴로 나타낼 수 있다.

$$
\begin{aligned}
X_1 - \mu_1 &= l_{11}F_1 + l_{12}F_2 + \cdots + l_{1d}F_d + \varepsilon_1 \\
X_2 - \mu_2 &= l_{21}F_1 + l_{22}F_2 + \cdots + l_{2d}F_d + \varepsilon_2 \\
&\vdots \\
X_p - \mu_p &= l_{p1}F_1 + l_{p2}F_2 + \cdots + l_{pd}F_d + \varepsilon_p
\end{aligned}
$$

$where\ \mu_i\ and\ l_{ij}\ are\ unknown\ constants, i = 1, \ldots, p, j = 1, \ldots, d$
$\varepsilon_i\ are\ unobserved\ error\ terms, i = 1, \ldots, p$

$$\underset{(p\times 1)}{\boldsymbol{X}} - \underset{(p\times 1)}{\boldsymbol{\mu}} = \underset{(p\times d)}{\boldsymbol{L}}\underset{(d\times 1)}{\boldsymbol{F}} + \underset{(p\times 1)}{\boldsymbol{\varepsilon}}$$

$$\begin{pmatrix} X_1 \\ X_2 \\ \vdots \\ X_p \end{pmatrix} - \begin{pmatrix} \mu_1 \\ \mu_2 \\ \vdots \\ \mu_p \end{pmatrix} = \begin{pmatrix} l_{11} & l_{12} & \cdots & l_{1d} \\ l_{21} & l_{22} & \cdots & l_{2d} \\ \vdots & \vdots & \ddots & \vdots \\ l_{p1} & l_{p2} & \cdots & l_{pd} \end{pmatrix} \begin{pmatrix} F_1 \\ F_2 \\ \vdots \\ F_d \end{pmatrix} + \begin{pmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_p \end{pmatrix}$$

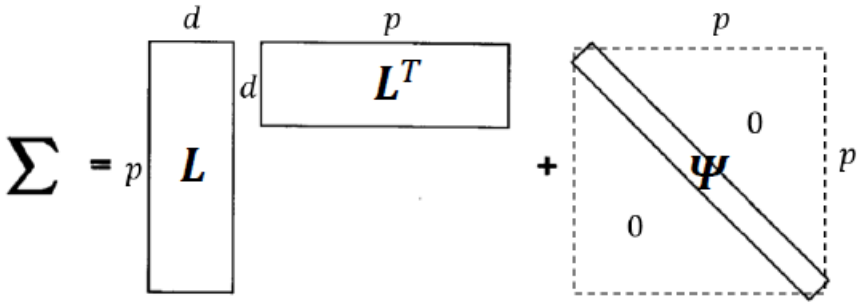
$where\ \mu_i\ and\ l_{ij}\ are\ unknown\ constants, i = 1, \dots, p, j = 1, \dots, d$
$\varepsilon_i\ are\ unobserved\ error\ terms, i = 1, \dots, p$

여기서 loading은 선형결합의 weight를 의미

- $\boldsymbol{F}$는 잠재변수, 즉 요인(Factor)벡터, $\boldsymbol{L}$은 factor loading의 행렬을 나타낸다.
  - ↳ $Cov(F_j, \varepsilon_i) = 0$
- 서로 독립인 $\boldsymbol{F}$ 와 $\boldsymbol{\varepsilon}$ 은 다음을 가정한다.

  각 잠재변수 Factor들, 잠재오차항들끼리도 각각 독립이라고 가정
  - $\boldsymbol{\varepsilon}_i$ 와 F_j 는 각각 $i = 1, \dots, p, j = 1, \dots, d$ 에 대해 independent한 random variable
  - $E(\boldsymbol{F}) = \boldsymbol{0},\quad E(\boldsymbol{\varepsilon}) = \boldsymbol{0},\quad Cov(\boldsymbol{F}) = \boldsymbol{I}_d,\quad Cov(\boldsymbol{\varepsilon}) = \boldsymbol{\Psi} = diag(\psi_1, \psi_2, \dots, \psi_p)$

Factor analysis : Model



$$E(X-\mu)=0$$

$$Cov(\boldsymbol{X}-\boldsymbol{\mu})=Cov(\boldsymbol{LF}+\boldsymbol{\varepsilon})=Cov(\boldsymbol{LF})+Cov(\boldsymbol{\varepsilon})=\boldsymbol{L}Cov(\boldsymbol{F})\boldsymbol{L}^T+Cov(\boldsymbol{\varepsilon})$$
$$=\boldsymbol{LL}^T+\boldsymbol{\Psi}$$

- Diagonal element

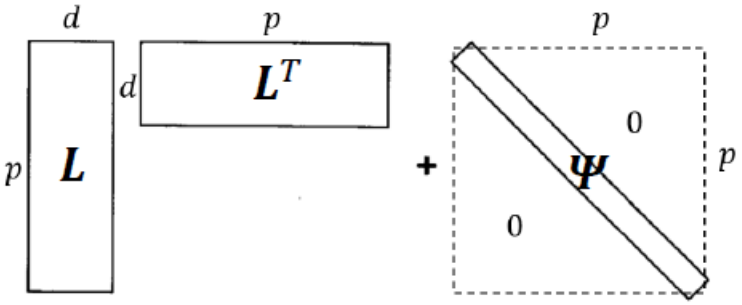
$$Var(X_i-\mu_i)=Var(X_i)=i^{th}\ diagonal\ element\ of\ (\boldsymbol{LL}^T+\boldsymbol{\Psi})$$
$$=l_{i1}^2+l_{i2}^2+\cdots+l_{id}^2+\psi_i$$
$$=h_i^2+Var(\varepsilon_i)$$

$\Rightarrow h_i^2=l_{i1}^2+l_{i2}^2+\cdots+l_{id}^2$: 공통요인분산(communality)

$\Rightarrow Var(\varepsilon_i)=\psi_i$: 특정분산(specific variance)

- 한 변수(X_i)의 분산 = 공통요인분산(h_i^2) + 특정분산(ψ_i)

Factor analysis : Model



$$E(X-\mu)=0$$

$$Cov(\boldsymbol{X}-\boldsymbol{\mu})=Cov(\boldsymbol{LF}+\boldsymbol{\varepsilon})=Cov(\boldsymbol{LF})+Cov(\boldsymbol{\varepsilon})=\boldsymbol{L}Cov(\boldsymbol{F})\boldsymbol{L}^T+Cov(\boldsymbol{\varepsilon})$$
$$=\boldsymbol{LL}^T+\boldsymbol{\Psi}$$

- Nondiagonal element

$$Cov(X_i-\mu_i,X_j-\mu_j)=Cov(X_i,X_j)=(i,j)^{th}\ element\ of\ (\boldsymbol{LL}^T+\boldsymbol{\Psi})$$
$$=l_{i1}l_{j1}+l_{i2}l_{j2}+\cdots+l_{id}l_{jd}+0$$
$$=l_{i1}l_{j1}+l_{i2}l_{j2}+\cdots+l_{id}l_{jd}$$

- $\boldsymbol{\Psi}$이 diagonal matrix이므로 loading matrix, $\boldsymbol{L}$ 에 의해서만 결정된다.

Estimation : Other methods

- 회귀계수를 추정하는 방법에 Least square, MLE 등 여러 방법이 있듯, 요인분석모형의 모수추정에도 Principal component method 외에 다른 방법들이 있다.

  ⇒ principal factor method, iterated principal factor method, maximum likelihood factor method

What to do?: L과 Ψ를 추정 ($X - \mu = LF + \epsilon$)

Estimation : Principal component method

- Principal Component Analysis와는 다름
- 하지만 공분산행렬의 eigenvector를 이용한다는 공통점
- 과정

① 관측된 데이터의 표본 공분산행렬, $\mathbf{S}$를 구하여 True 공분산행렬 $\boldsymbol{\Sigma} = \boldsymbol{LL}^T + \boldsymbol{\Psi}$를 추정.

② $\mathbf{S}$를 spectral decomposition한 결과를 이용해 $\hat{\boldsymbol{L}}$ 추정.

③ ②에서 구한 $\hat{\boldsymbol{L}}$을 이용하여 $\boldsymbol{\Psi}$ 추정.

$\widehat{\Psi_i} = (i,i)th\ element\ of\ \left(\mathbf{S} - \hat{\boldsymbol{L}}\hat{\boldsymbol{L}}^{\boldsymbol{T}}\right)$로 계산

Estimation : Principal component method

STEP① 표본 공분산행렬, $\mathbf{S}$를 계산하여 True 공분산행렬 $\boldsymbol{\Sigma} = \boldsymbol{L}\boldsymbol{L}^T + \boldsymbol{\Psi}$ 추정

$$\mathbf{S} \cong \boldsymbol{\Sigma} = \boldsymbol{L}\boldsymbol{L}^T + \boldsymbol{\Psi}$$

STEP② $\mathbf{S}$를 spectral decomposition한 결과를 이용해 $\hat{\boldsymbol{L}}$ 추정.

$\mathbf{S}$를 spectral decomposition (eigen-decomposition)

$\mathbf{S}$는 대칭행렬이므로 $\mathbf{C}^{\mathrm{T}} = \mathbf{C}^{-1}$

$$\mathbf{S} = \mathbf{C}\mathbf{D}\mathbf{C}^{\mathrm{T}} = \begin{pmatrix} \boldsymbol{v}_1 & \cdots & \boldsymbol{v}_p \end{pmatrix} \begin{pmatrix} \lambda_1 & & 0 \\ & \ddots & \\ 0 & & \lambda_p \end{pmatrix} \begin{pmatrix} \boldsymbol{v}_1{}' \\ \\ \boldsymbol{v}_p{}' \end{pmatrix}$$

Estimation : Principal component method

$$\mathbf{S} = \mathbf{CDC}^{\mathrm{T}} = \begin{pmatrix} \boldsymbol{v}_1 & \cdots & \boldsymbol{v}_p \end{pmatrix} \begin{pmatrix} \lambda_1 & & 0 \\ & \ddots & \\ 0 & & \lambda_p \end{pmatrix} \begin{pmatrix} \boldsymbol{v}_1{}' \\ \\ \boldsymbol{v}_p{}' \end{pmatrix}$$

$$= \begin{pmatrix} \boldsymbol{v}_1 & \cdots & \boldsymbol{v}_p \end{pmatrix} \begin{pmatrix} \sqrt{\lambda_1} & & 0 \\ & \ddots & \\ 0 & & \sqrt{\lambda_p} \end{pmatrix} \begin{pmatrix} \sqrt{\lambda_1} & & 0 \\ & \ddots & \\ 0 & & \sqrt{\lambda_p} \end{pmatrix} \begin{pmatrix} \boldsymbol{v}_1{}' \\ \vdots \\ \boldsymbol{v}_p{}' \end{pmatrix}$$

$$= \mathbf{CD}^{\frac{1}{2}}\mathbf{D}^{\frac{1}{2}}\mathbf{C}^{\mathrm{T}} = \mathbf{CD}^{\frac{1}{2}}\left(\mathbf{CD}^{\frac{1}{2}}\right)^{\mathrm{T}}$$

근데 $\mathbf{CD}^{\frac{1}{2}}$ 은 $p \times p$ 행렬이고, ②에서 우리가 추정하고자 하는 $\boldsymbol{L}$은 $p \times d$ 행렬 어떡하지?

⇒ p개의 변수에 대한 $d(< p)$개 factor의 weight, loading이기 때문에

Estimation : Principal component method

- 공분산행렬의 eigenvalue는 해당 eigenvector가 데이터에 갖는 설명력을 의미.
- Spectral decomposition한 후, eigenvalue가 큰 순서대로 d개의 eigenvector를 남기고, 이를 이용하여 다음과 같은 행렬을 만든다. $v_{d+1}, \dots, v_p$는 버린다.

$$\underset{(p\times d)(d\times p)}{\hat{\boldsymbol{L}}\,\hat{\boldsymbol{L}}^{T}} = \begin{pmatrix} \boldsymbol{v}_1 & \cdots & \boldsymbol{v}_d \end{pmatrix} \begin{pmatrix} \sqrt{\lambda_1} & & 0 \\ & \ddots & \\ 0 & & \sqrt{\lambda_d} \end{pmatrix} \begin{pmatrix} \sqrt{\lambda_1} & & 0 \\ & \ddots & \\ 0 & & \sqrt{\lambda_d} \end{pmatrix} \begin{pmatrix} \boldsymbol{v}_1{}' \\ \vdots \\ \boldsymbol{v}_d{}' \end{pmatrix}$$

$$= \boldsymbol{C}_0\boldsymbol{D}_0{}^{\frac{1}{2}}\boldsymbol{D}_0{}^{\frac{1}{2}}\boldsymbol{C}_0{}^{\mathrm{T}} = \underset{(p\times d)}{\boldsymbol{C}_0\boldsymbol{D}_0{}^{\frac{1}{2}}}\underset{(d\times p)}{\left(\boldsymbol{C}_0\boldsymbol{D}_0{}^{\frac{1}{2}}\right)^{\mathrm{T}}}$$

Estimation : Principal component method

STEP③ ②에서 구한 $\widehat{\boldsymbol{L}}$을 이용하여 $\boldsymbol{\Psi}$ 추정.

$$\underset{(p\times d)(d\times p)}{\widehat{\boldsymbol{L}}\,\widehat{\boldsymbol{L}}^{T}} = \begin{pmatrix} \boldsymbol{v}_1 & \cdots & \boldsymbol{v}_d \end{pmatrix} \begin{pmatrix} \sqrt{\lambda_1} & & 0 \\ & \ddots & \\ 0 & & \sqrt{\lambda_d} \end{pmatrix} \begin{pmatrix} \sqrt{\lambda_1} & & 0 \\ & \ddots & \\ 0 & & \sqrt{\lambda_d} \end{pmatrix} \begin{pmatrix} \boldsymbol{v}_1{}' \\ \vdots \\ \boldsymbol{v}_d{}' \end{pmatrix}$$

$$= \boldsymbol{C}_0\boldsymbol{D}_0{}^{\frac{1}{2}}\boldsymbol{D}_0{}^{\frac{1}{2}}\boldsymbol{C}_0{}^{\mathrm{T}} = \underset{(p\times d)}{\boldsymbol{C}_0\boldsymbol{D}_0{}^{\frac{1}{2}}}\underset{(d\times p)}{\left(\boldsymbol{C}_0\boldsymbol{D}_0{}^{\frac{1}{2}}\right)^{\mathrm{T}}}$$

- $Cov(\boldsymbol{X}) = \boldsymbol{L}\boldsymbol{L}^{T} + \boldsymbol{\Psi}$ 이므로, $\widehat{\Psi_i} = (i,i)th\ element\ of\ \left(\boldsymbol{S} - \widehat{\boldsymbol{L}}\widehat{\boldsymbol{L}}^{\boldsymbol{T}}\right)$로 추정.

  ⇒ 이는 STEP②에서 $\widehat{\boldsymbol{L}}\widehat{\boldsymbol{L}}^{\boldsymbol{T}}$ 도출 시 버린 $v_{d+1}, \dots, v_p$의 부분을 $\widehat{\boldsymbol{\Psi}}$로 일부 반영해주는 것으로 볼 수 있다.

왜 일부 반영해주었다고 했을까? 생각해보자!

???


여긴
어디지?!
난
누구지!!

Example : Perception data

- 7명의 서로 다른 사람에 대하여 5개의 성격특징에 대한 점수를 매기게 하였다.

  Kind, Intelligent, Happy, Likeable, Just

	X_1	X_2	X_3	X_4	X_5
People	Kind	Intelligent	Happy	Likeable	Just
FSM1[a]	1	5	5	1	1
SISTER	8	9	7	9	8
FSM2	9	8	9	9	8
FATHER	9	9	9	9	9
TEACHER	1	9	1	1	9
MSM[b]	9	7	7	9	9
FSM3	9	7	9	9	7

[a] Female schoolmate 1.
[b] Male schoolmate.

Example : Perception data

STEP① 표본 공분산행렬, $\mathbf{S}$를 계산하여 True 공분산행렬 $\boldsymbol{\Sigma} = \boldsymbol{LL}^T + \boldsymbol{\Psi}$ 추정

물론 분석을 수행하는 연구자의 주관에 달린 결정이지만, 이 example을 수행한 사람은 각 변수가 성격특징에 관한 점수이고 단위의 차이가 없음에도 불구하고 표본 상관계수행렬을 사용

왜일까..

$$\mathbf{R} = \begin{pmatrix} 1.000 & .296 & \mathbf{.881} & \mathbf{.995} & .545 \\ .296 & 1.000 & -.022 & .326 & \mathbf{.837} \\ \mathbf{.881} & -.022 & 1.000 & \mathbf{.867} & .130 \\ \mathbf{.995} & .326 & \mathbf{.867} & 1.000 & .544 \\ .545 & \mathbf{.837} & .130 & .544 & 1.000 \end{pmatrix}$$

- 변수 1,3,4 그리고 변수 2,5가 높은 상관관계를 보이는 것을 알 수 있다.

  ⇒ 요인분석의 결과도 이와 유사한 결과가 도출될 것으로 예상할 수 있음

Example : Perception data

STEP② 공분산행렬 $\boldsymbol{S}$를 spectral decomposition한 결과를 이용해 $\hat{\boldsymbol{L}}$ 추정.

- 표본 상관관계행렬 $\boldsymbol{R}$의 eigenvalue는 $3.263, 1.538, 0.168, 0.031, 0$
- 2개의 잠재변수, factor를 이용하여 모형을 세우기로 결정.
- Eigenvalue 3.263와 1.538에 해당하는 eigenvector는 다음과 같다.

$$\lambda_1 = 3.263 \quad \boldsymbol{v}_1 = \begin{pmatrix} 0.537 \\ 0.288 \\ 0.434 \\ 0.537 \\ 0.390 \end{pmatrix} \qquad \lambda_2 = 1.538 \quad \boldsymbol{v}_2 = \begin{pmatrix} -0.186 \\ 0.651 \\ -0.473 \\ -0.169 \\ 0.538 \end{pmatrix}$$

Example : Perception data

STEP② 공분산행렬 $\mathbf{S}$를 spectral decomposition한 결과를 이용해 $\hat{\boldsymbol{L}}$ 추정.

$$\hat{\boldsymbol{L}} = \left(\begin{array}{c|c} 0.537 & -0.186 \\ 0.288 & 0.651 \\ 0.434 & -0.473 \\ 0.537 & -0.169 \\ 0.390 & 0.538 \end{array}\right)\begin{pmatrix} \sqrt{3.263} & 0 \\ 0 & \sqrt{1.538} \end{pmatrix} = \begin{pmatrix} 0.969 & -0.231 \\ 0.519 & 0.807 \\ 0.785 & -0.587 \\ 0.971 & -0.210 \\ 0.704 & 0.667 \end{pmatrix}$$

$$\hat{\boldsymbol{L}}\hat{\boldsymbol{L}}^{T} = \begin{pmatrix} 0.993 & 0.317 & 0.896 & 0.990 & 0.528 \\ 0.317 & 0.921 & -0.066 & 0.335 & 0.904 \\ 0.896 & -0.066 & 0.960 & 0.885 & 0.161 \\ 0.990 & 0.335 & 0.885 & 0.987 & 0.543 \\ 0.528 & 0.904 & 0.161 & 0.543 & 0.940 \end{pmatrix}$$

Example : Perception data

STEP③ ②에서 구한 $\widehat{\boldsymbol{L}}$을 이용하여 $\boldsymbol{\Psi}$ 추정, $\widehat{\Psi_i} = (i,i)th\ element\ of\ \left(\mathbf{S} - \widehat{\boldsymbol{L}}\widehat{\boldsymbol{L}}^{\boldsymbol{T}}\right)$

$$diagonal\ elements\ of\ \widehat{\boldsymbol{\Psi}} = diagonal\ elements\ of\ \left(\boldsymbol{S} - \widehat{\boldsymbol{L}}\widehat{\boldsymbol{L}}^{T}\right)$$

$$nondiagonal\ elements\ of\ \widehat{\boldsymbol{\Psi}} = 0$$

$$\widehat{\boldsymbol{\Psi}} = \begin{pmatrix} 0.007 & 0 & 0 & 0 & 0 \\ 0 & 0.079 & 0 & 0 & 0 \\ 0 & 0 & 0.040 & 0 & 0 \\ 0 & 0 & 0 & 0.013 & 0 \\ 0 & 0 & 0 & 0 & 0.060 \end{pmatrix}$$

Example : Perception data

- Principal component method로 parameter를 추정한 결과는 다음과 같다.

$$\hat{\boldsymbol{L}} = \begin{pmatrix} 0.969 & -0.231 \\ 0.519 & 0.807 \\ 0.785 & -0.587 \\ 0.971 & -0.210 \\ 0.704 & 0.667 \end{pmatrix} \qquad \hat{\boldsymbol{\Psi}} = \begin{pmatrix} 0.007 & 0 & 0 & 0 & 0 \\ 0 & 0.079 & 0 & 0 & 0 \\ 0 & 0 & 0.040 & 0 & 0 \\ 0 & 0 & 0 & 0.013 & 0 \\ 0 & 0 & 0 & 0 & 0.060 \end{pmatrix}$$

$$\begin{aligned} Kind - \mu_{kind} &= 0.969 Factor_1 - 0.231 Factor_2 + \varepsilon_1 \\ Intelligent - \mu_{intelligent} &= 0.519 Factor_1 + 0.807 Factor_2 + \varepsilon_2 \\ Happy - \mu_{happy} &= 0.785 Factor_1 - 0.587 Factor_2 + \varepsilon_3 \\ &\vdots \end{aligned}$$

- 변수 1,3,4 와 2,5가 각각 유사하긴 하지만, 요인들의 의미를 해석하는 것이 쉽지 않다.

  의미 해석이 용이한 loading이 나오도록 이를 개선할 수 있는 방법이 있을까?

Factor rotation : Background

- $E(\boldsymbol{X}-\boldsymbol{\mu})=\boldsymbol{0}$: 예측변수 편차의 기댓값이 0
- 그럼 앞의 model setting 하에서 예측변수들의 covariance는?

$$Cov(\boldsymbol{X}-\boldsymbol{\mu})=Cov(\boldsymbol{LF}+\boldsymbol{\varepsilon})=Cov(\boldsymbol{LF})+Cov(\boldsymbol{\varepsilon})=\boldsymbol{L}Cov(\boldsymbol{F})\boldsymbol{L}^T+Cov(\boldsymbol{\varepsilon})$$
$$=\boldsymbol{L}\boldsymbol{L}^T+\boldsymbol{\Psi}$$

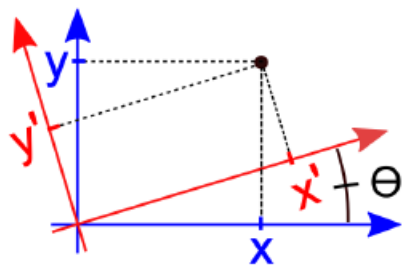
- 그런데 $QQ^T=Q^TQ=I$를 만족하는 행렬 Q에 대하여, $\boldsymbol{L}^*=\boldsymbol{LQ}$, $\boldsymbol{F}^*=\boldsymbol{Q}^T\boldsymbol{F}$ 로 두면, 새로운 loading L^*와 새로운 factor F^*에 대해서도 다음이 만족.

$$Cov(\boldsymbol{X}-\boldsymbol{\mu})=Cov(\boldsymbol{L}^*\boldsymbol{F}^*+\boldsymbol{\varepsilon})=Cov(\boldsymbol{L}^*\boldsymbol{F}^*)+Cov(\boldsymbol{\varepsilon})=\boldsymbol{L}^*Cov(\boldsymbol{F}^*)\boldsymbol{L}^{*T}+Cov(\boldsymbol{\varepsilon})$$
$$=\boldsymbol{L}^*\boldsymbol{L}^{*T}+\boldsymbol{\Psi}=\boldsymbol{L}(\boldsymbol{Q}\boldsymbol{Q}^T)\boldsymbol{L}^T+\boldsymbol{\Psi}=\boldsymbol{L}\boldsymbol{L}^T+\boldsymbol{\Psi}$$

$\Rightarrow$ 이는 어떤 의미가 있을까?

Note : Matrix representation of rotation

- 앞에 소개한 내용은 축을 가만히 두고 한 벡터를 회전시키는 것.
- 하지만, 우리의 관심사는 벡터를 가만히 두고 축을 회전시키는 것.



- 축을 θ°만큼 회전시키는 것은, 해당 벡터 $\boldsymbol{x}$ 를 반대방향으로 θ°만큼,
  즉 벡터 $\boldsymbol{x}$ 를 $-\theta^{\circ}$만큼 회전시키는 것과 같다.
- 회전변환의 역행렬은 transpose와 같으므로, 축을 θ° 회전시킨 후의 벡터 $\boldsymbol{x}^{*}$는

$$\boldsymbol{x}^{*} = \begin{pmatrix} cos\,\theta & -sin\,\theta \\ sin\,\theta & cos\,\theta \end{pmatrix}^{-1} \cdot \boldsymbol{x} = \begin{pmatrix} cos\,\theta & -sin\,\theta \\ sin\,\theta & cos\,\theta \end{pmatrix}^{T} \cdot \boldsymbol{x} = \begin{pmatrix} cos\,\theta & sin\,\theta \\ -sin\,\theta & cos\,\theta \end{pmatrix} \cdot \boldsymbol{x}$$

Rotation of factors and their loadings

- 예측변수의 편차($\boldsymbol{X}-\boldsymbol{\mu}$)는 잠재변수, 즉 요인들의 선형결합($\boldsymbol{LF}$)과 오차($\boldsymbol{\varepsilon}$)의 합
- d개의 요인을 나타내는 요인벡터를 회전변환하더라도($\boldsymbol{F} \rightarrow \boldsymbol{F}^*$), 선형결합의 loading을 그에 따라 잘 바꿔준다면, 모형의 covariance structure는 동일하다.

$$Cov(\boldsymbol{X}-\boldsymbol{\mu}) = Cov(\boldsymbol{LF}+\boldsymbol{\varepsilon}) = \boldsymbol{LL}^T + \boldsymbol{\Psi}$$
$$= Cov(\boldsymbol{L}^*\boldsymbol{F}^*+\boldsymbol{\varepsilon}) = \boldsymbol{L}(\boldsymbol{QQ}^T)\boldsymbol{L}^T + \boldsymbol{\Psi} = \boldsymbol{LL}^T + \boldsymbol{\Psi}$$

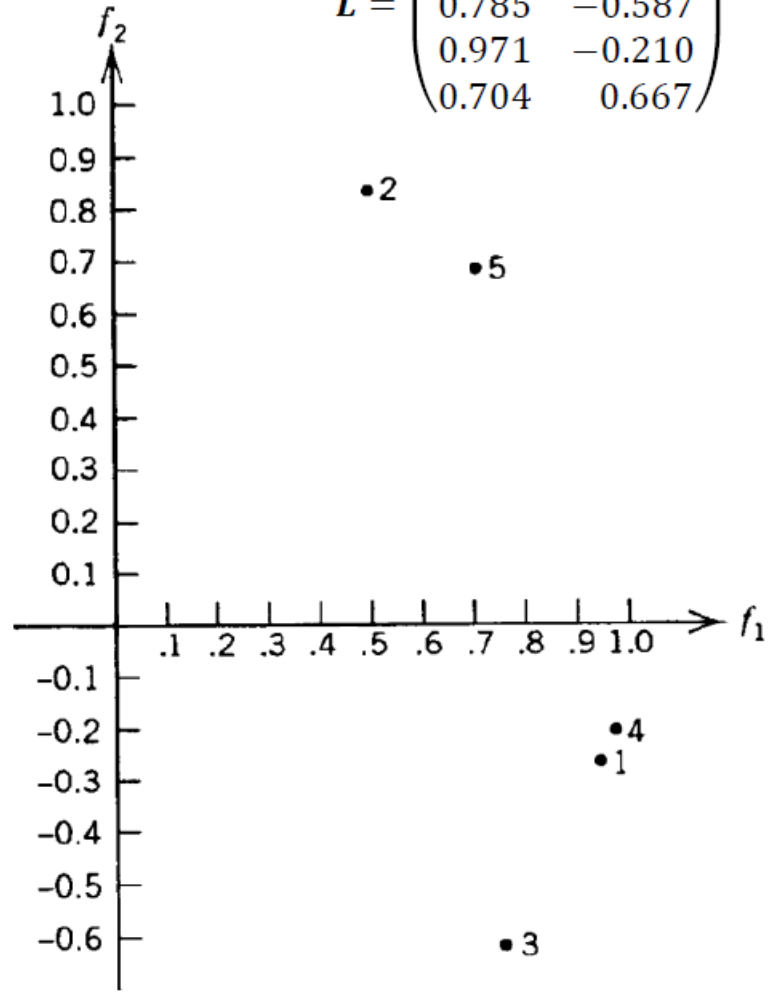
$\Rightarrow$ 즉, 어떤 예측변수들을 설명하는 요인들의 조합은 유일하지 않다.

$\Rightarrow$ 반대로 얘기하면, 요인들의 해석이 더 용이하도록 요인들을 회전시킬 수 있다.

Example : Perception data

- 5개의 변수의 2개의 factor에 대한 가중치, 즉 loading을 F_1, F_2평면에 나타낸 것은 다음과 같다.
- loading의 절댓값이 전체적으로 0이나 1에 가까우면 해석이 용이하다고 할 수 있다.
- 앞에서 소개한 factor의 회전변환으로 어떻게 이를 달성할 수 있을까? 그리고 무엇을 기준으로 최적의 회전변환을 판단해야 할까?

$$\hat{L} = \begin{pmatrix} 0.969 & -0.231 \\ 0.519 & 0.807 \\ 0.785 & -0.587 \\ 0.971 & -0.210 \\ 0.704 & 0.667 \end{pmatrix}$$



Example : Perception data

- 요인의 회전변환을 통해 해석에 용이한 요인을 찾는 작업은 오른쪽 그림으로 쉽게 이해할 수 있다.
- 축을 θ°만큼 회전변환한 새로운 요인 벡터

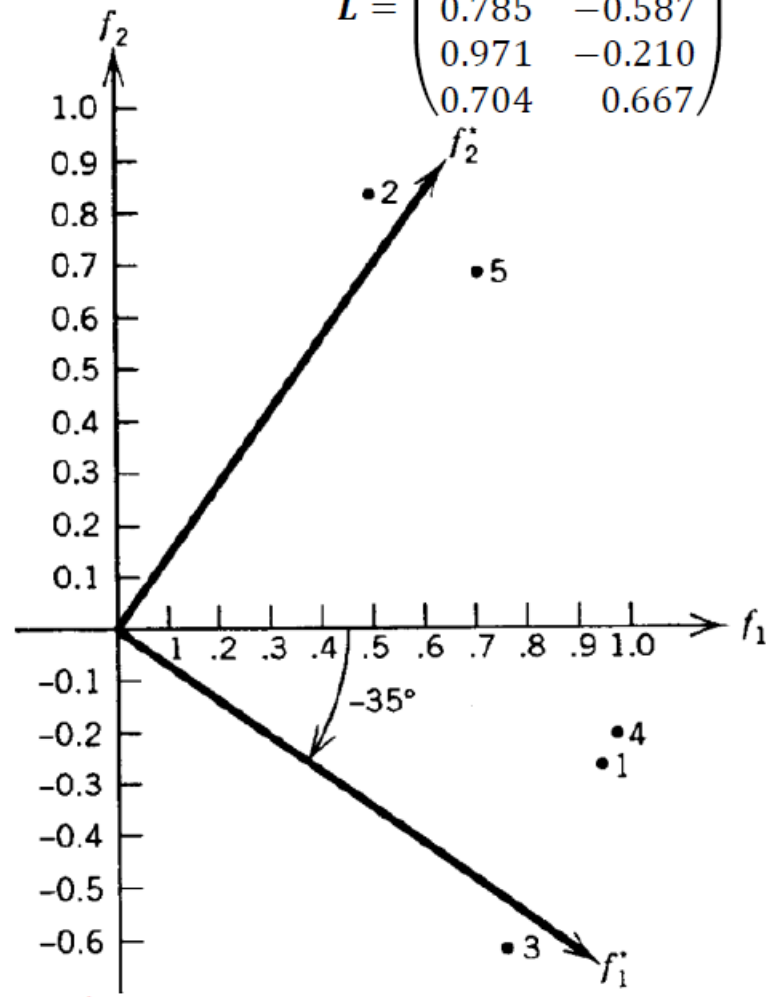
$$\boldsymbol{F}^{*} = \boldsymbol{Q}^{-1}\boldsymbol{F} = \boldsymbol{Q}^{T}\boldsymbol{F} = \begin{pmatrix} cos\theta & sin\theta \\ -sin\theta & cos\theta \end{pmatrix}\boldsymbol{F}$$

- 그에 따른 새로운 loading estimate

$$\hat{\boldsymbol{L}}^{*} = \hat{\boldsymbol{L}}\boldsymbol{Q} = \hat{\boldsymbol{L}}\begin{pmatrix} cos\theta & -sin\theta \\ sin\theta & cos\theta \end{pmatrix}$$

회전 후 loading인 $\hat{\boldsymbol{L}}$는 왜 $\boldsymbol{L}$의 오른쪽에 $\boldsymbol{Q}$를 곱할까?
⇒어떤 행렬 $\boldsymbol{X}$에 대해 $\hat{\boldsymbol{L}}\boldsymbol{F} = \hat{\boldsymbol{L}}\boldsymbol{X}\boldsymbol{Q}^{T}\boldsymbol{F}$ 가 만족하려면 $\boldsymbol{X} = \boldsymbol{Q}$ 여야하기 때문

$$\hat{\boldsymbol{L}} = \begin{pmatrix} 0.969 & -0.231 \\ 0.519 & 0.807 \\ 0.785 & -0.587 \\ 0.971 & -0.210 \\ 0.704 & 0.667 \end{pmatrix}$$



Example : Perception data

요인들의 계수가 좀더 해석이 용이해졌다.
근데 이게 요인의 최적 회전변환일까?

Variables	Principal Component Loadings		Graphically Rotated Loadings		앞에 나왔던 공통요인분산 Communalities,
	f_1	f_2	f_1	f_2	$\hat{h}_i^2$
Kind	.969	−.231	**.927**	.367	.993
Intelligent	.519	.807	−.037	**.959**	.921
Happy	.785	−.587	**.980**	−.031	.960
Likeable	.971	−.210	**.916**	.385	.987
Just	.704	.667	.194	**.950**	.940

$$\hat{\boldsymbol{L}}^* = \hat{\boldsymbol{L}}\boldsymbol{Q} = \hat{\boldsymbol{L}}\begin{pmatrix}\cos(-35^\circ) & -sin(-35^\circ)\\ sin(-35^\circ) & cos(-35^\circ)\end{pmatrix} = \begin{pmatrix}0.969 & -0.231\\ 0.519 & 0.807\\ 0.785 & -0.587\\ 0.971 & -0.210\\ 0.704 & 0.667\end{pmatrix}\begin{pmatrix}0.819 & 0.574\\ -0.574 & 0.819\end{pmatrix} = \begin{pmatrix}0.927 & 0.367\\ -0.037 & 0.959\\ 0.980 & -0.031\\ 0.916 & -0.385\\ 0.194 & 0.950\end{pmatrix}$$

Example : Perception data

$$\hat{\boldsymbol{L}} = \begin{pmatrix} 0.969 & -0.231 \\ 0.519 & 0.807 \\ 0.785 & -0.587 \\ 0.971 & -0.210 \\ 0.704 & 0.667 \end{pmatrix}$$

- 추정된 loading matrix($\hat{\boldsymbol{L}}$)의 각 loading 값들이 최대한 서로 달랐으면 좋겠다
- Varimax rotation : 다음 V를 최대화하는 θ°만큼 회전하자

$$V = \sum_{j=1}^{d} \left\{ \frac{1}{p} \sum_{i=1}^{p} Y_{ij}^2 - \frac{1}{p^2} \left(\sum_{i=1}^{p} Y_{ij} \right)^2 \right\}, \qquad where\ Y_{ij} = \frac{\hat{l}_{ij}^2}{\hat{h}_i^2} = \frac{\hat{l}_{ij}^2}{\hat{l}_{i1}^2 + \hat{l}_{i2}^2 + \cdots + \hat{l}_{id}^2}$$

제곱의 평균 - 평균의 제곱, 즉 분산 꼴

X_j변수의 공통요인분산 중,
j번째 factor에서 온 영향의 비율

- 일일이 1°씩 factor축을 돌려가며 해볼 수는 없기 때문에, 이 작업은 컴퓨터 소프트웨어를 통해 수행한다.

Example : Perception data

Variables	Principal Component Loadings		Graphically Rotated Loadings		Varimax Rotated Loadings		Communalities
	f_1	f_2	f_1	f_2	f_1	f_2	$\hat{h}_i^2$
Kind	.969	−.231	**.927**	.367	**.951**	.298	.993
Intelligent	.519	.807	−.037	**.959**	.033	**.959**	.921
Happy	.785	−.587	**.980**	−.031	**.975**	−.103	.960
Likeable	.971	−.210	**.916**	.385	**.941**	.317	.987
Just	.704	.667	.194	**.950**	.263	**.933**	.940

Rotation 하지 않았을 때의 loading들보다 해석이 더 용이하다

Summary of FA (PC method)

1. X변수들이 잠재변수의 선형결합으로 이뤄졌다 가정$(X - \mu = LF + \epsilon)$

2. d개의 PC를 통해 L과 Ψ를 추정 $(Cov(X) = LL' + \Psi)$

3. 회전변환을 통해 계수 조정

PCA vs FA

PCA	FA
비모수적	모수적
분산중심	공분산중심
X들의 선형결합으로 PC를 표현	F들의 선형결합으로 X를 표현
PC 사용	PC가 아닌 다른 추정 방법 가능

질문 받아요

Reference

[1] Jaejoon Lee (Yonsei, ESC), (2018) “*Dimension Reduction*” ppt

[2] Christopher Bishop, (2006), “*Pattern Recognition and Machine Learning*”

[3] Sunjoo Kim(Yonsei), (2019), “*Computer Vision*” lecture note

[4] Kwangeun Seol, (2018), “*Seoulkwangeun.jpg*” 존잘 사진

감사합니다