2024년 2학기

선형대수

12장

Chapter 12

- SECTION 12-1 고유값과 고유벡터의 해석
- SECTION 12-2 고유값 구하기
- SECTION 12-3 고유벡터 찾기
- SECTION 12-4 정방 행렬의 대각화
- SECTION 12-5 대칭 행렬의 특별함
- SECTION 12-6 특이 행렬의 고유값 분해
- SECTION 12-7 이차식, 정부호성 및 고유값
- SECTION 12-8 일반화된 고유값 분해

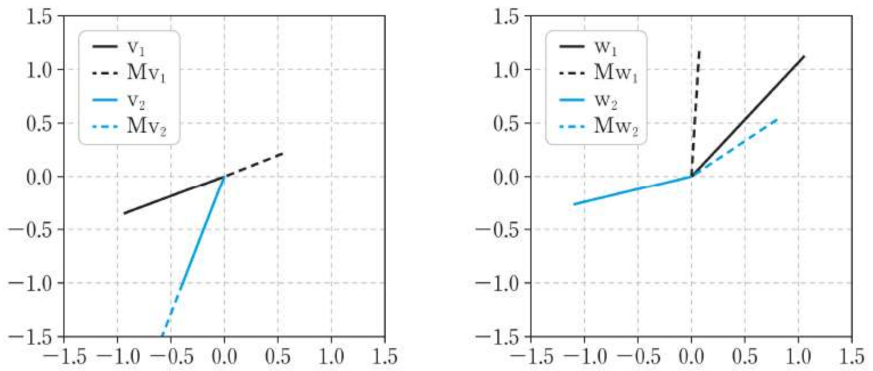
SCH

SECTION 12-1 고유값과 고유벡터의 해석(1)

12.1.1 기하학

- 2×2 행렬로 곱하기 전과 후의 벡터
  - 왼쪽 그림의 두 벡터(v_1과 v_2)는 고유벡터, 오른쪽 그림의 두 벡터는 고유벡터가 아님
  - 고유벡터는 행렬 벡터 곱셈이 스칼라-벡터 곱셈처럼 작동한다는 것을 의미
    - 고유값은 정방 행렬에 대해서만 정의됨

고유값 방정식(eigenvalue eq.): $\mathbf{A}\mathbf{v} = \lambda\mathbf{v}$

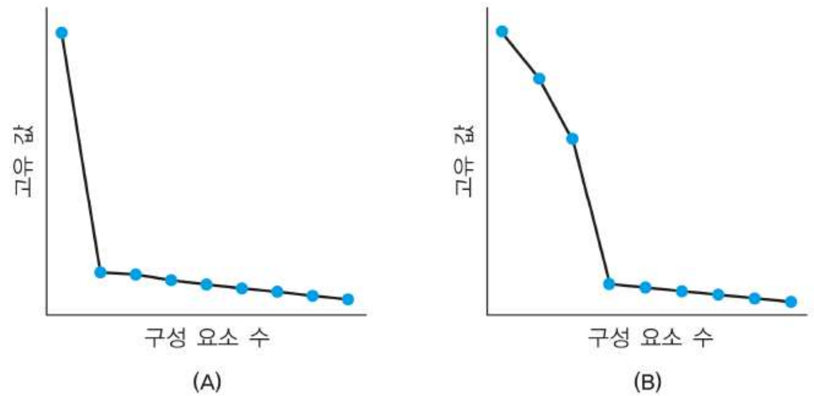


고유벡터의 기하학적 그림

SECTION 12-1 고유값과 고유벡터의 해석(2)

12.1.2 통계(주성분 분석)

- 통계를 적용하는 이유 중 하나는 변수 간의 관계를 파악하고 정량화하기
- 가설
  - 암호화폐공간 전체가 단일 시스템으로 운영되는지(모든 코인의 가치가 함께 오르내림을 의미) 아니면 해당 공간 내에 독립적인 하위 분류가 존재하는지(일부 코인 또는 코인 그룹은 다른 코인의 가치와 별개로 변동됨을 의미)를 파악
- 테스트
  - 시간 경과에 따른 다양한 암호화폐의 가격이 포함된 데이터 집합에 대해 주성분 분석을 수행



다변량 데이터 집합의 시뮬레이션된 스크리 도표
(데이터 집합의 공분산 행렬의 고유값 그래프)

SECTION 12-1 고유값과 고유벡터의 해석(3)

12.1.3 잡음 감쇠

- 데이터 집합에는 잡음(noise)이 포함
  - 잡음은 설명할 수 없거나(예를 들어 무작위 변동) 원치 않는(예를 들어 무선 신호의 전기선 잡음 요소) 데이터 집합의 분산을 의미
  - 무작위 잡음을 줄이는 한 가지 방법은 시스템의 고유값과 고유벡터를 식별한 다음 작은 고유값과 관련된 데이터 공간에서 방향을 '투영'
    - 즉 무작위 잡음이 전체분산에 기여하는 바가 상대적으로 작다고 가정
    - 데이터 차원을 '투영'한다는 것은 임계값보다 낮은 일부 고유값을 0으로 설정한 후 데이터 집합을 재구성하는 것을 의미

분산?

- 평균, 편차, 분산, 표준편차
  - 편차: 측정 값과 평균의 차이 값
  - 분산: 편차의 제곱의 평균 값, 변량들이 퍼져 있는 정도를 의미
  - 표준편차: 분산은 수치가 커서, 제곱근으로 줄여 사용함 (분산은 표준편차의 제곱)
    - 표준편차가 크면? 관측 값들이 들쭉날쭉하다는 의미
    - 표준편차가 작으면? 관측 값들이 대체로 유사

	1	2	3	4	5
관측 값	175	177	179	181	183
평균	(175+177+179+181+183) / 5 = 179				
편차 (관측 – 평균)	-4	-2	0	2	4
편차제곱	16	4	0	4	16
분산	(16+4+0+4+16) / 5 = 8				
표준편차	$\sqrt{8} \fallingdotseq 2.828$				

SECTION 12-1 고유값과 고유벡터의 해석(4)

12.1.4 차원 축소(데이터 압축)

- 압축(compress)
  - 데이터 품질에 미치는 영향을 최소화하면서 데이터의 크기(바이트 단위)를 줄이는 것
  - 먼저 고유값 분해를 수행한 다음, 데이터 공간의 작은 방향과 연관된 고유값과 고유벡터를 삭제하고 상대적으로 큰 고유벡터와 고유값 쌍만 전송
    - 데이터 집합을 데이터의 가장 중요한 특성을 나타내는 기준 벡터 집합으로 분해한 다음 원본 데이터의 고품질 형태로 재구성

SECTION 12-2 고유값 구하기(1)

- 정방 행렬의 고유값 분해 – 고유값을 찾은 다음 각 고유값을 사용하여 해당 고유벡터를 찾음
- 파이썬 코드로 고유값 찾기

```
A = np.array([[1,2],[3,4]])
evals = np.linalg.eig(A)
print(evals)
print(np.around(evals[0], 2))
```

- 두 개의 고유값(소수점 둘째까지 반올림)은 -0.37과 5.37

SECTION 12-2 고유값 구하기(2)

- 행렬의 고유값 구하기

$$\mathrm{A}\,\mathrm{v} = \lambda\mathrm{v}$$ ← 고유값 방정식

$$\mathrm{A}\,\mathrm{v} - \lambda\mathrm{v} = 0$$ ← 우변을 빼서 식을 영벡터로 만들기

$$(\mathrm{A} - \lambda\mathrm{I})\mathrm{v} = 0$$ ← 행렬을 λ만큼 이동하면서 공통된 인수 v를 추출

- 행렬을 미지의 고유값 λ로 이동하고 행렬의 행렬식을 0으로 설정한 다음 λ를 구하는 것이 고유값 찾기의 핵심
  - 선형대수학에서는 단순한 해를 무시하므로 v=0을 고유벡터로 보지 않음
  - 고유값에 의해 이동된 행렬이 특이 행렬이라는 것을 의미, 특이 행렬만이 자명하지 않은(nontrivial) 널 공간을 갖기 때문임
    - 행렬 $\tilde{\mathrm{A}}\mathrm{v} = 0$일때 $\tilde{\mathrm{A}}$ 가 가역이라 $\tilde{\mathrm{A}}^{-1}$ 이 존재할 경우: $\tilde{\mathrm{A}}\mathrm{v} = 0$은 오직 자명해 v=0임
      - 자명해(trivial solution)은 너무 당연하게 보이는 해, 2x=0, x=0
    - 행렬 $\tilde{\mathrm{A}}\mathrm{v} = 0$일때 $\tilde{\mathrm{A}}$ 가 비가역이면 $\tilde{\mathrm{A}}^{-1}$ 이 존재하지 않음. 이경우 v=0 이외의 해를 가짐

$$\tilde{\mathrm{A}} = \mathrm{A} - \lambda\mathrm{I}$$
$$\tilde{\mathrm{A}}\mathrm{v} = 0$$

$$|\mathrm{A} - \lambda\mathrm{I}| = 0$$

SECTION 12-2 고유값 구하기(3)

- 2 × 2 행렬에서 고유값 찾기

$$|\mathbf{A} - \lambda \mathbf{I}| = \mathbf{0}$$

$$\left| \begin{bmatrix} a & b \\ c & d \end{bmatrix} - \lambda \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \right| = 0$$

$$\begin{vmatrix} a-\lambda & b \\ c & d-\lambda \end{vmatrix} = 0$$

$$(a-\lambda)(d-\lambda) - bc = 0$$

$$\lambda^2 - (a+d)\lambda + (ad-bc) = 0$$

- 고유값 구하기 개념의 논리적 과정
  - 행렬-벡터 곱셈은 스칼라-벡터 곱셈(고유값 방정식)처럼 동작
  - 고유값 방정식을 영벡터로 설정하고 공통항을 추출
  - 이렇게 하면 고유벡터가 고유값에 의해 이동된 행렬의 널 공간에 있음을 알 수 있음. 영벡터를 고유벡터로 간주하지 않으므로 이동된 행렬은 특이 행렬
  - 따라서 이동된 행렬의 행렬식을 0으로 설정하고 미지의 고유값을 구함
- 행렬의 특성 다항식(characteristic polynomial) - 고유값으로 이동된 행렬의 행렬식을 0으로 둔 것
  - 때문에 MxM 행렬은 M개의 고유값을 갖게됨

SECTION 12-3 고유벡터 찾기(1)

- 고유벡터를 찾는 파이썬 코드

```
A = np.array([[1,2],[3,4]])
evals, evecs = np.linalg.eig(A)

for i in range(len(evals)):
  print(np.around(evals[i], 2), np.around(evecs[i], 2))
```

```
-0.37 [-0.82 -0.42]
5.37 [ 0.57 -0.91]
```

- 고유벡터는 행렬 evecs의 열에 존재하며 고유값과 같은 순서
- 고유벡터는 열벡터!

실습 문제

- 행렬 A의 고유값과 고유벡터를 구하고, 증명하세요
  - 증명1: 고유값 방정식: $\mathrm{A}\mathbf{v} = \lambda\mathbf{v}$
  - 증명2: $|\mathrm{A} - \lambda\mathrm{I}| = 0$

```
A = np.arange(4).reshape(2,2)
```

SECTION 12-3 고유벡터 찾기(2)

- 고유벡터를 찾는 방법
  - λ만큼 이동한 행렬의 널 공간에 있는 벡터 v를 구하는 것

$$\mathbf{v}_i \in \mathrm{N}(\mathrm{A} - \lambda_i \mathrm{I})$$

  - 행렬의 고유값

$$\begin{bmatrix} 1 & 2 \\ 2 & 1 \end{bmatrix} \Rightarrow \lambda_1 = 3,\ \lambda_2 = -1$$

□ 행렬의 고유벡터를 구하기 위해 행렬을 3만큼 이동하고 그 널 공간에서 벡터를 찾음

$$\begin{bmatrix} 1-3 & 2 \\ 2 & 1-3 \end{bmatrix} = \begin{bmatrix} -2 & 2 \\ 2 & -2 \end{bmatrix} \Rightarrow \begin{bmatrix} -2 & 2 \\ 2 & -2 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

SECTION 12-3 고유벡터 찾기(3)

12.3.1 고유벡터의 부호와 크기 불확정성

- v가 행렬의 고유벡터라면 0을 제외한 모든 실수값 α에 대해 αv도 고유벡터
  - 이동된 행렬을 다시 살펴봤을 때 [1 1]이 널 공간에 대한 유일한 기저벡터는 아님
  - [4 4] 또는 [-5.4 -5.4] 등 무수한 벡터가 될 수 있음
  - 벡터 [1 1]의 크기를 조정한 모든 벡터는 널 공간의 기저가 될 수 있음
  - 고유벡터는 크기가 아닌 방향 때문에 중요

- 가능한 널 공간 기저벡터가 무한대
  - 단 하나의 '최상의' 기저벡터가 존재하는가?
    - '최상의' 기저벡터라는건 없지만 단위 정규화된 고유벡터(유클리드 노름 1)가 있음
  - 고유벡터의 '올바른' 부호는 무엇인가?
    - 없음

SECTION 12-4 정방 행렬의 대각화(1)

- 고유값 방정식에는 하나의 고유값과 하나의 고유벡터만 존재
  - M x M 행렬에는 M개의 고유값 방정식이 있음

$$\begin{aligned} A\mathbf{v}_1 &= \lambda_1 \mathbf{v}_1 \\ &\vdots \\ A\mathbf{v}_M &= \lambda_M \mathbf{v}_M \end{aligned}$$

- 행렬 방정식으로 변환
  - 고유벡터 행렬의 각 열이 정확히 하나의 고유값으로 크기가 조정
  - 대각 행렬을 뒤에서 곱하여 구현

$$\begin{bmatrix} @ & @ & @ \\ @ & @ & @ \\ @ & @ & @ \end{bmatrix} = \begin{bmatrix} v_{11} & v_{21} & v_{31} \\ v_{12} & v_{22} & v_{32} \\ v_{13} & v_{23} & v_{33} \end{bmatrix} = \begin{bmatrix} v_{11} & v_{21} & v_{31} \\ v_{12} & v_{22} & v_{32} \\ v_{13} & v_{23} & v_{33} \end{bmatrix} \begin{bmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \\ 0 & 0 & \lambda_3 \end{bmatrix}$$

$$= \begin{bmatrix} \lambda_1 v_{11} & \lambda_2 v_{21} & \lambda_3 v_{31} \\ \lambda_1 v_{12} & \lambda_2 v_{22} & \lambda_3 v_{32} \\ \lambda_1 v_{13} & \lambda_2 v_{23} & \lambda_3 v_{33} \end{bmatrix}$$

SECTION 12-4 정방 행렬의 대각화(2)

- 행렬 고유값 방정식(정방 행렬의 대각화)

$$\mathrm{AV} = \mathrm{V}\Lambda$$

- NumPy의 eig 함수는 행렬의 고유벡터와 벡터의 고유값을 반환

```
A = np.random.randn(2,2)
evals, V = np.linalg.eig(A)
L = np.diag(evals)
np.around(A@V - V@L, 2)
```

$$\mathrm{AV} = \mathrm{V}\Lambda$$
$$\mathrm{A} = \mathrm{V}\Lambda\mathrm{V}^{-1}$$
$$\Lambda = \mathrm{V}^{-1}\mathrm{AV}$$