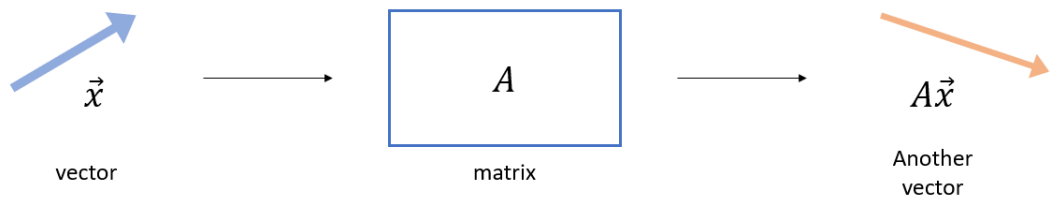
리드오프 3주차 전반부

고유값과 고유 벡터

1-1. 개념과 기하학적 의미

행렬은 선형 변환 연산임을 2주차 후반부에서 확인했었다. 무엇을 변환시켜 준 것일까? 행렬은 벡터를 변환시켜 다른 벡터를 출력해준다.



즉 행렬을 이용해 벡터를 변환 시켜 주면 변환 후의 벡터($A\vec{x}$)는 변환 전의 벡터($\vec{x}$)와 비교했을 때, 크기도 방향도 모두 변할 수 있다는 것이다. 그런데 특정한 벡터와 행렬은 선형 변환을 취해주었을 때, 크기만 바뀌고 방향은 바뀌지 않을 수도 있다.

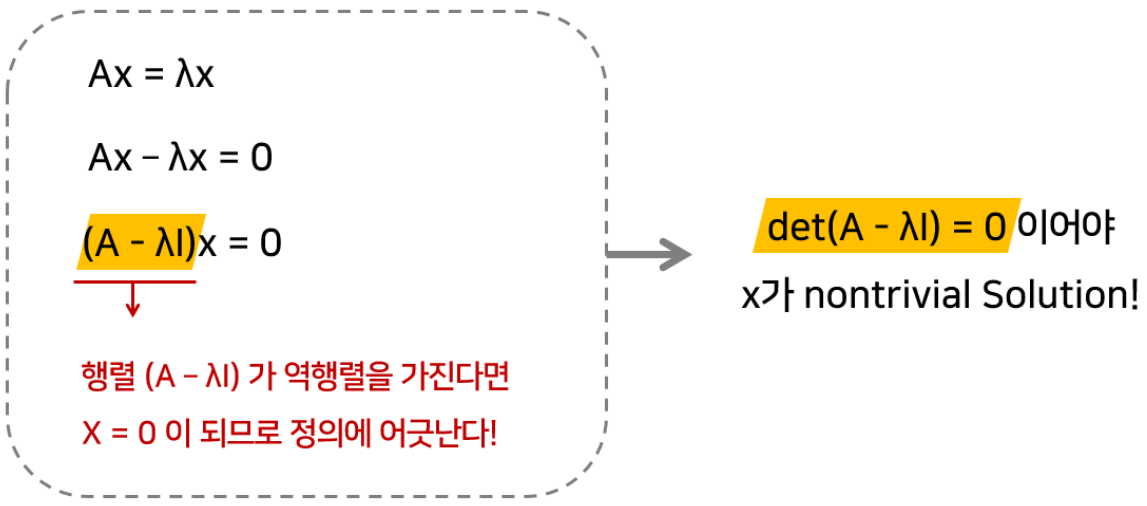
$$\begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 3 \\ 3 \end{bmatrix}$$

즉, 입력 벡터($\vec{x}$)를 A 로 선형변환 시킨 결과($A\vec{x}$) 가 상수배라는 것이고 이는 $Ax = \lambda x$ 로 표현된다.

고유값과 고유벡터

A가 n×n 행렬일 때, 만약 Ax=λx 가 0이 아닌 벡터 x가 존재한다면 스칼라 λ를 A의 고유값(eigenvalue)이라 한다.
λ가 A의 고유값이면, Ax=λx 인 0이 아닌 모든 벡터는 λ에 대응하는 A의 고유벡터(eigenvector)이다.

위 정의 속 등식 $Ax = \lambda x$ 을 $(A - \lambda I)x = 0$ 로 바꾸어 계산할 수 있는데, x가 0인 경우는 자명하고(trivial) 그것은 우리의 관심사가 아니기 때문에 자명하지 않은(nontrivial) x를 가지도록 다음과 같이 고유값 λ를 구하게 된다.



1-2. 중요성

그래서 이 고유값이 왜 중요한 것인지, 한번 짚고 넘어가보자. 이를 위해 잠시 대각행렬(Diagonal Matrix)의 개념을 불러오려 한다. 만약 A라는 행렬이 $\begin{bmatrix} -1 & 0 \\ 0 & 2 \end{bmatrix}$라고 해보자. 2주차 후반부의 개념이 잘 이해되었다면, 이는 표준 기저벡터들을 x축 방향으로는 -1만큼, y축 방향으로는 2만큼 스케일링하는 선형변환임을 알 수 있을 것이다. 그리고 고유벡터의 개념을 생각해보면, 이 표준 기저벡터들은 고유벡터가 되기도 함을 알 수 있다. 또 -1, 2 가 각각 고유값이 될 것이다.

대각행렬이 좋은 이유 중 하나는 행렬을 아무리 곱해도, 무슨 일이 일어날지 계산하기 편하다는 것이다. 예를 들어, A^3x 는 $\begin{bmatrix} (-1)^3 & 0 \\ 0 & 2^3 \end{bmatrix} x$ 이다. (A 변환을 3번 곱하는 상황을 기하학적으로도 한번 상상해보자) 이는 달리 생각하면, 기저 벡터를 특정 값으로 스케일링하는 것뿐임을 의미한다. (즉 A 행렬에서는 표준 기저벡터들이 고유 벡터가 된다) 행렬을 100번 곱해도, 표준 기저 벡터를 고유값의 100제곱으로 스케일링하는 변환과 동일하다. 대각 행렬이 아닌 일반적인 행렬을 100제곱하는 연산과 비교해보면, 너무나 간단한 연산이다.

그러면 우리는 다음 질문으로 넘어가게 된다. 일반적인 상황에서도 이러한 고유 벡터들을 기저 벡터로써 사용할 수 있다면?

우리는 전에서 이미 행렬이 기저벡터를 변환시키는 변환임을 관찰한 바 있다. 표준기저벡터를 원하는 벡터(즉 고유벡터)로 변환시킨 뒤, 고유값만큼 스케일링 후 다시 역행렬(바뀐 기저벡터를 다시 표준기저벡터로 돌리는 변환)을 적용한다면 이는 여전히 같은 변환이다. 식으로 나타낸다면 $A = V^{-1}\Lambda V$이다. 물론 모든 상황에서 고유벡터가 존재하지는 않겠지만, 존재한다면 매우 편한 연산이 가능해질 것이다. 그리고 이러한 변환은 단순히 행렬의 거듭제곱뿐 아니라, 역행렬과 determinant 계산, trace 계산에서도 엄청난 계산의 이점을 지니게 된다. 이것이 대각화(diagonalization) 및 고유값 분해(eigen decomposition)가 등장하게 된 배경이라고도 할 수 있다.

1) 행렬식

$$\begin{aligned} det(A) = det(V\Lambda V^{-1}) = det(V)det(\Lambda)det(V^{-1}) \\ = det(\Lambda) = \lambda_1\lambda_2\cdots\lambda_n \end{aligned}$$

2) 거듭제곱

$$\begin{aligned} A^k = (V\Lambda V^{-1})^k = V\Lambda V^{-1}V\Lambda V^{-1} \\ = V\Lambda^k V^{-1} \end{aligned}$$

3)대각합

$$\begin{aligned} tr(A) = tr(V\Lambda V^{-1})^k = tr(V^{-1}V\Lambda) \\ = tr(\Lambda) = \lambda_1\lambda_2\cdots\lambda_n \end{aligned}$$

1-3. 고유값 분해

어떤 $n \times n$차원 행렬 A가 다음과 같이 구성되어 있다고 생각해보자. 여기서 $\lambda_i \text{ for } i = 1, 2, \cdots, n$ 은 임의의 실수이다.

$$A = \begin{bmatrix} | & | & & | \\ \lambda_1 a_1 & \lambda_2 a_2 & \cdots & \lambda_n a_n \\ | & | & & | \end{bmatrix} \in \mathbb{R}^{n\times n} \quad \cdots (1)$$

그렇다면 행렬 A는 다음과 같이 인수분해 할 수 있다.

$$A = \begin{bmatrix} | & | & & | \\ a_1 & a_2 & \cdots & a_n \\ | & | & & | \end{bmatrix} \begin{bmatrix} \lambda_1 & 0 & \cdots & 0 \\ 0 & \lambda_2 & 0 & 0 \\ & & \ddots & \\ 0 & 0 & \cdots & \lambda_n \end{bmatrix} \quad \cdots (2)$$

임의의 행렬 A에 대해 여러 개의 고윳값과 고유벡터를 얻었다고 해보자. 다시 말해, 다음 식을 만족하는 고윳값과 고유벡터 n개씩 획득할 수 있고, 또 고유 벡터 행렬은 다음과 같이 표현할 수 있다.

$$Av_i = \lambda_i v_i \text{ for } i = 1, 2, \cdots, n$$

$$V = \begin{bmatrix} | & | & & | \\ v_1 & v_2 & \cdots & v_n \\ | & | & & | \end{bmatrix}$$

$$AV = \begin{bmatrix} | & | & & | \\ \lambda_1 v_1 & \lambda_2 v_2 & \cdots & \lambda_n v_n \\ | & | & & | \end{bmatrix}$$

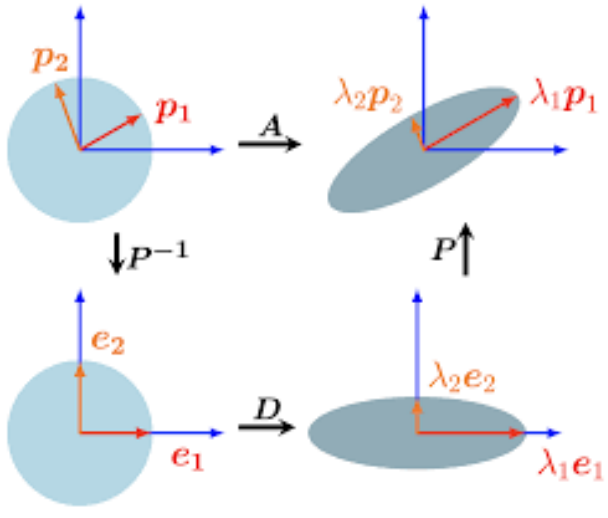
또 고윳값들을 대각성분에 모아둔 행렬 $\Lambda \in \mathbb{R}^{n\times n}$을 아래와 같이 생각해본다면,

$$\Lambda = \begin{bmatrix} \lambda_1 & 0 & \cdots & 0 \\ 0 & \lambda_2 & 0 & 0 \\ & & \ddots & \\ 0 & 0 & \cdots & \lambda_n \end{bmatrix}$$

와 같다. (1)과 (2)의 식을 이용하고, 또 만약 모든 고유벡터들이 선형독립이라면

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

와 같이 A를 표현할 수 있다. 이것이 고유값 분해이다. 그리고 이는 **기존의 선형변환을 '돌리기','늘리기','돌리기'의 세 과정으로 분해해서 생각할 수도 있다.**



📌 언제나 고유값 분해가 가능할까?

모든 경우에 고유값 분해를 할 수 있는 것은 아니다. 정방행렬 A가 고유값 분해가 가능하려면, 행렬 A의 열벡터가 모두 **선형독립**이어야 한다. 이는 $n \times n$ 의 행렬 A에서 고유값 n 개를 찾으려 할 때, 선형독립이지 않은 벡터들 사이에선 n개의 고유값를 찾는 것이 불가능하기 때문이다.

1-4. 대칭 행렬(Symmetric Matrix)의 고유값 분해

1) 정의

$A = A^T$의 성질을 만족하는 행렬을 대칭행렬이라고 한다. 만약 행렬 A가 방금의 예시처럼 고유값분해할 수 있다면

$$A = V\Lambda V^{-1} = A^T = (V\Lambda V^{-1})^T = (V^{-1})^T \Lambda^T V^T$$

Λ는 대각행렬이므로 $\Lambda^T = \Lambda$이고, $(V^{-1})^T = (V^T)^{-1}$이므로,

$$V\Lambda V^{-1} = (V^T)^{-1}\Lambda V^T \Rightarrow V^T = V^{-1} \Rightarrow VV^T = V^T V = I$$

따라서 대칭 행렬은 $A = V\Lambda V^T$와 같이 분해할 수 있다. 대칭행렬은 고유값 분해에서 특수한 경우이므로 $A = Q\Lambda Q^T$ 라고도 표기한다. 또 증명은 생략하겠지만, 대칭행렬은 항상 고유값 분해가 가능하며 직교행렬로 대각화가 가능함을 기억하자.

📌 **정리**

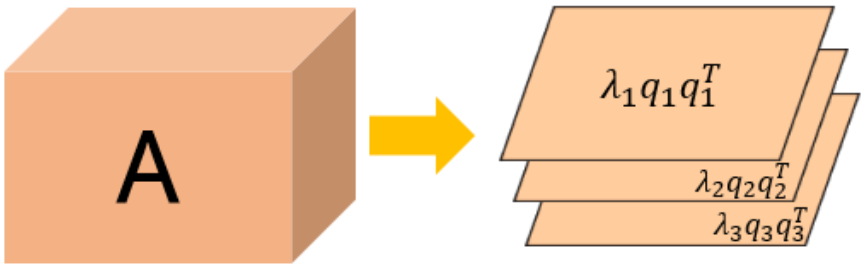
대칭행렬은 (1) 항상 고유값 분해가 가능하며 (2) **직교행렬로 고유값 분해**가 가능하다.

2) 대칭행렬 고유값 분해의 성질

대칭행렬을 고유값 분해하면 매우 재미있는 성질을 발견할 수 있다. A가 3×3 대칭행렬이고, 이를 고유값 분해하여 자세히 살펴본다고 할 때,

$$\begin{aligned} A &= \begin{bmatrix} | & | & | \\ q_1 & q_2 & q_3 \\ | & | & | \end{bmatrix} \begin{bmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \\ 0 & 0 & \lambda_n \end{bmatrix} \begin{bmatrix} - & q_1^T & - \\ - & q_2^T & - \\ - & q_3^T & - \end{bmatrix} \\ &= \begin{bmatrix} \lambda_1 q_1 & \lambda_2 q_2 & \lambda_3 q_3 \end{bmatrix} \begin{bmatrix} q_1^T \\ q_2^T \\ q_3^T \end{bmatrix} \\ &= \lambda_1 q_1 q_1^T + \lambda_2 q_2 q_2^T + \lambda_3 q_3 q_3^T \end{aligned}$$

와 같은 추가적인 분해가 가능해진다. $q_i q_i^T$ 가 Rank1 matrix인 것을 고려하면, A 행렬을 크레이프케익처럼 슬라이스로 쪼개어 볼 수 있다는 것이다. (적절한 비유인지는 모르겠습니다.. ㅎ)



조금 더 살펴보자면

$\begin{bmatrix}2\\1\\3\end{bmatrix} = 2\begin{bmatrix}1\\0\\0\end{bmatrix} + 1\begin{bmatrix}0\\1\\0\end{bmatrix} + 3\begin{bmatrix}0\\0\\1\end{bmatrix}$ 으로 볼 수 있는 것처럼, 대칭행렬 A에 대해 어떠한 Ax 벡터도 $\lambda_1 q_1 q_1^T x + \lambda_2 q_2 q_2^T x + \lambda_3 q_3 q_3^T x$ 로 볼 수 있게 된다는 것이다. $q_i^T x$ 가 벡터의 내적인 것을 고려하면, 즉 위 식은 다음과 같이 표현할 수 있다.

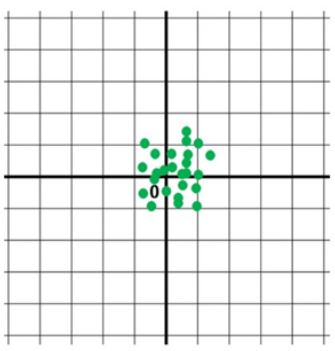
> 만약 A가 symmetric하다면, Ax는 x를 A의 고유벡터로 분해하고 그걸 고유값만큼 scaling한 후, 다시 더하는 것

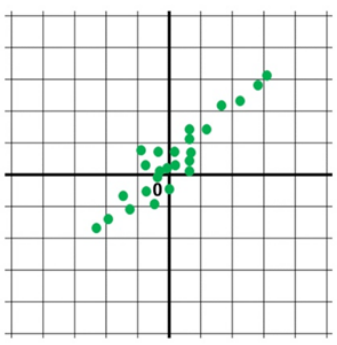
이는 곧 고유값이 큰 변환만을 살려, 데이터 압축에도 응용할 수 있게 됨을 의미한다. 예를 들어 λ_1이 전체 고유값의 95%를 차지하고 있다면, 한 개만을 살려도 원래의 변환과 유사하게 만들 수 있다는 것이다. 곧 PCA 또는 사진 압축의 원리가 된다. 고유값 분해의 예시는 아니지만, 아래의 수식을 통해 그 흐름을 짐작해볼 수 있을 것이다.

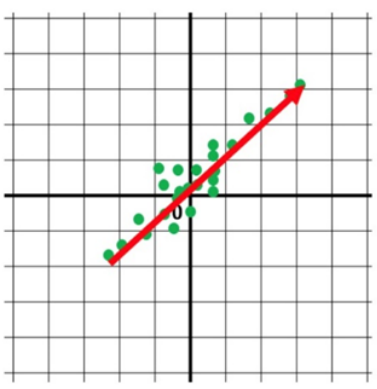
$$\begin{bmatrix}2\\1\\25\end{bmatrix} = 2\begin{bmatrix}1\\0\\0\end{bmatrix} + 1\begin{bmatrix}0\\1\\0\end{bmatrix} + 25\begin{bmatrix}0\\0\\1\end{bmatrix} \simeq 25\begin{bmatrix}0\\0\\1\end{bmatrix}$$

PCA(Primary Component Analysis)

그럼 이제 고유값 분해의 가장 중요한 응용 중 하나인 PCA를 살펴보자. 그 전에 이전 시간에 보았던 공분산행렬을 다시 불러올 필요가 있다.

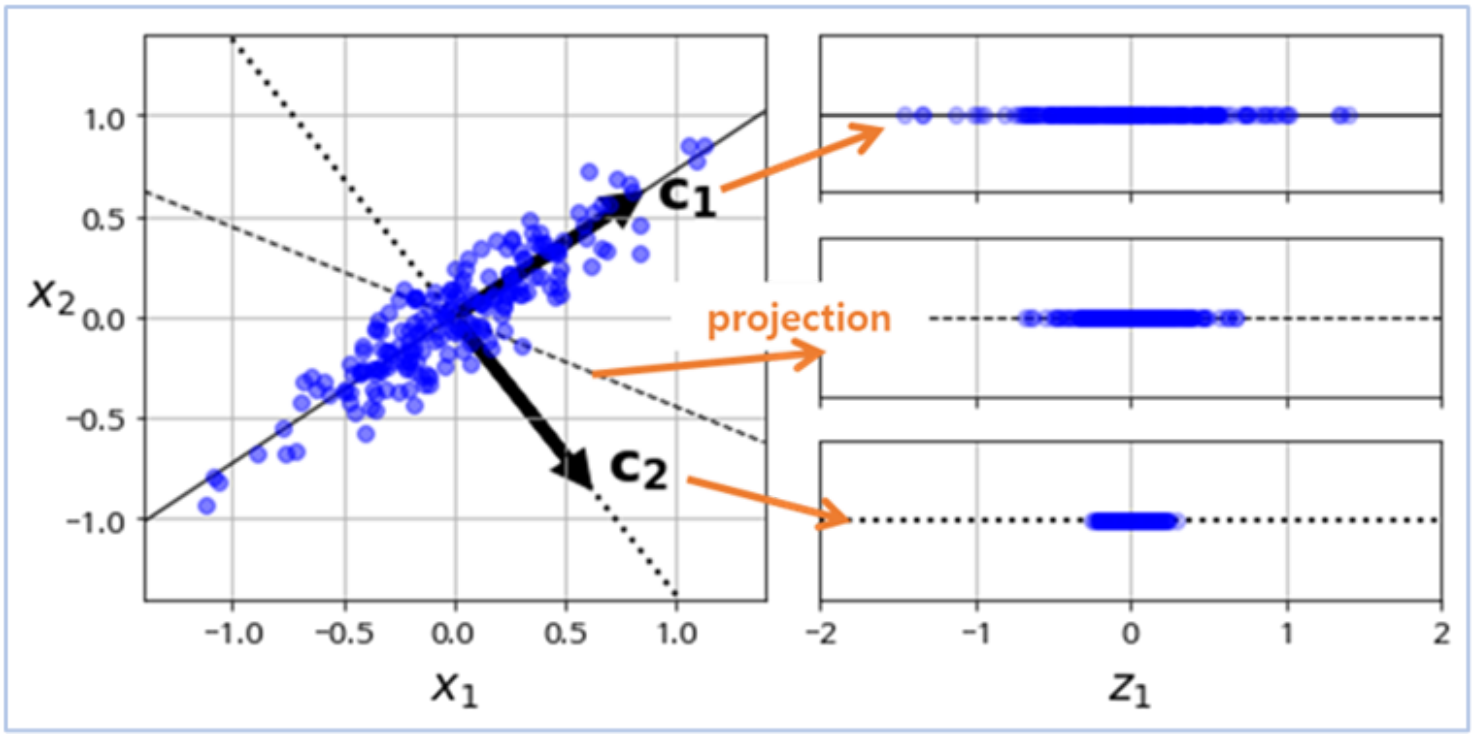






Gaussian distribution을 통해 랜덤하게 분포된 선형 변환 전 데이터의 분포가 $\begin{bmatrix}3 & 2\\2 & 4\end{bmatrix}$ 라는 공분산 행렬에 의해 변환되었다.

이 데이터에서 우리는 차원을 축소시키고자 한다. 간단하게 2차원 데이터로 생각해보았을 때, 2개의 변수에 대해 하나의 축으로 Projection 하고 싶다는 것인데, 어떤 선을 선택해야 가장 많은 정보를 보존할 수 있을까? 바로 분산이 가장 큰 방향이다.



🤔 **왜 분산이 가장 큰 방향으로 Projection하는 걸까?**

데이터의 차원 축소는 정보의 손실을 동반한다. 그렇기 때문에, 아무리 차원의 저주가 심각하더라도 중요한 정보는 보존할 필요가 있다. 그렇다면 정보가 많고 적음은 어떻게 알 수 있을까? 통계적데이터마이닝 수업에서 김재직 교수님의 설명을 빌려오자면.. 만약 열 x_1이 눈의 개수라고 해보자. 눈의 개수는 대부분 2개이기 때문에, 분산은 0에 가까울 것이다. 이때 우리는 이 변수가 분류 문제에서든, 회귀 문제에서든 중요한 정보를 갖고 있지는 않을 것이라는 것을 직관적으로 느낄 수 있다. 그러므로 항상 정답인 것은 아니지만, 분산이 큰 축 또는 변수가 이러한 문제에서 좋은 정보를 갖고 있을 것이라고 판단할 수 있다.

후에 나올 증명에서, 라그랑주 승수법의 λ와 고유벡터의 개념에서 나오는 λ가 혼용되어 헷갈릴 수도 있을 것 같습니다. 그렇지만 이 λ의 값이 중요한 것은 아니고, "이렇게 수식을 풀었더니, 고유벡터의 조건을 만족하는 식이 나오더라~" 라는 흐름으로 받아주시면 좋을 것 같습니다~!

내적 값 역시 정사영한 벡터의 크기에 의해 영향을 받으므로, 정사영 대상 벡터의 크기를 제한해 정사영 대상으로는 길이가 1인 단위 벡터를 생각하게 된다. 정사영 대상인 임의의 단위 벡터 $\vec{e}$를 생각해보자. d차원의 데이터를 $\vec{e}$ 에 정사영하여 1차원으로 차원을 축소하고 싶다. X를 $\vec{e}$에 대해 사영한 벡터는 $(X\vec{e})\vec{e}$ 이므로, 크기는 $X\vec{e}$ 가 된다.

계산의 편의를 위해 X의 각 열 평균은 0이라고 가정하고, $\vec{e}$에 정사영 된 데이터의 variance를 계산하면 다음과 같다.

$$Var(X\vec{e}) = \frac{1}{n}\sum_{i=1}^{n}(X\vec{e} - E(X\vec{e}))^2 \Rightarrow \frac{1}{n}\sum_{i=1}^{n}(X\vec{e})^2$$

$$Var(X\vec{e}) = \frac{1}{n}(X\vec{e})^T(X\vec{e})$$

$$= \frac{1}{n}\vec{e}^T X^T X\vec{e} = \frac{1}{n}\vec{e}^T(X^T X)\vec{e}$$

$$= \vec{e}^T\left(\frac{X^T X}{n}\right)\vec{e} \;=\; \vec{e}^T\Sigma\vec{e}$$

분산을 최대화하는 것이 목적이므로, 다음과 같은 최적화 문제를 생각할 수 있다.

$$max \quad \vec{e}^T\Sigma\vec{e}$$
$$subject\ to\ ||\vec{e}||^2 = 1$$

제약식이 있는 최적화 문제를 쉽게 해결하기 위해 우리는 라그랑주 승수법을 통해 제약이 없는 최적화 문제로 만들어줄 수 있다는 것을 배웠고,

$$L = \vec{e}^T\Sigma\vec{e} - \lambda(|\vec{e}|^2 - 1)$$

라그랑주 승수법을 통해 제약이 없는 문제가 되었으므로, Gradient가 0이 되는 지점을 활용해 최적 해를 구할 수 있다.

$$\frac{\partial L}{\partial \vec{e}} = 2\Sigma\vec{e} - 2\lambda\vec{e} = 0$$

–

$$\Sigma\vec{e^\star} = \frac{X^TX}{n}\vec{e} = \lambda\vec{e^\star}$$

Σ 가 행렬, 즉 선형 변환임을 고려하면, 최적의 값을 갖게 하는 벡터가 Σ 에 대해 고유벡터의 성질을 만족하는 벡터임을 볼 수 있다. 따라서 고유벡터에 정사영하는 것이 최대한의 정보를 보존할 수 있는 것이고, 위 결과를 이용해 계산하면 고유벡터에 정사영 했을 때 분산은 고유값임을 알 수 있다.

$$Var(X\vec{e^\star}) = \vec{e^\star}^T\Sigma\vec{e^\star} = \vec{e^\star}^T\lambda\vec{e^\star} = \lambda\vec{e^\star}^T\vec{e^\star} = \lambda$$

Additional Topic - PCA 추가 논의

여기까지만 해도 매우 뜻깊은 논의였던 것 같은데요.. 그러나 아직까지 조금 찜찜함이 남습니다. 고유벡터는 여러 개가 존재할 수 있었던 것 같은데, 단순히 저렇게 구하는 것만으로는 정말 최적 해라고 단정할 수 없지 않을까요? 그리고 PCA를 접해본 적이 있다면, 왜 그 다음으로 분산을 가장 잘 설명하는 방향이 첫 고유벡터와 직교였을까요? 다음의 증명을 통해 PCA에 대해 완벽하게 파헤쳐보도록 합시다.

아쉽게도 이 최적화 문제는 Convex Problem은 아니기 때문에, 좀 더 엄밀한 논의가 필요하다. 따라서 대칭 행렬의 고유값 분해를 다시 살펴볼 필요가 있다. 우선 우리는 대칭 행렬 A에 대해, 이를 $\lambda_1 q_1 q_1^T + \lambda_2 q_2 q_2^T + \cdots + \lambda_n q_n q_n^T$ 로 분해할 수 있음을 배웠고 q는 또 직교 벡터라고 했다. 따라서

$$\vec{e}^T\Sigma\vec{e} = \vec{e}^T\left(\lambda_1 q_1 q_1^T + \lambda_2 q_2 q_2^T + \cdots\right)\vec{e}$$

이다. 최적 값을 갖는 $\vec{e}$의 후보군은 임계점을 갖게 하는 q_i 중 결정될 것이다.

만약 고유값을 크기 순서대로 $\lambda_1, \lambda_2, \cdots$로 배열한 것이라면, $\vec{e}$ 가 q_1일 때 $[q_i^Tq_i = 1, (\text{단위벡터}),\ \ q_i^Tq_j(i \neq j) = 0\ (q\text{끼리는 직교})]$

$$q_1^T\left(\lambda_1 q_1 q_1^T + \lambda_2 q_2 q_2^T + \cdots\right)q_1 = q_1^T\left(\lambda_1 q_1 q_1^T + \lambda_2 q_2 q_2^T + \cdots\right)q_1 = \lambda_1 q_1^T q_1 q_1^T q + 0 + \cdots = \lambda_1$$

λ_1의 값을 가지며 최대가 된다는 것을 알 수 있다. 비로소 우리는 고유값이 가장 큰 벡터의 방향으로 정사영하는 것이 가장 큰 분산을 보존할 수 있다는 것을 알게 된다. 또 그렇게 PC1에 의한 분산의 설명량을 빼면, 남은 분산을 가장 잘 설명하는 것은 자연스럽게 그 다음으로 고유값이 큰 λ_2 가 될 것이다.

또한 $\Sigma = \lambda_1 q_1 q_1^T + \lambda_2 q_2 q_2^T + \cdots + \lambda_d q_d q_d^T$를 이용하면 고유값을 모두 더한 총합이 공분산 행렬에 의해 설명되는 분산이라는 것도 알 수 있을 것이다. 따라서 $\frac{\sum_{j=1}^{m}\lambda_j}{\sum_{i=1}^{d}\lambda_i}$ 와 같은 분산 설명 비율을 통해 PC의 개수를 결정한다는 것도 참고로 알아두자.