선형대수학 정리

기석

https://www.edx.org/course/linear-algebra-foundations-frontiers-utaustinx-ut-5-05x

목차

- 8주차 꼭 듣기!!

- 12주차 꼭 듣기!!

목차

오늘 수업 시간에 질문

1. 선형대수학을 이걸 왜 하는가?
   - machine learning

2. Rank-1 update의 메커니즘은 무엇인가??? LU factorization에서 쓰이는데 잘 모르겠다.

cT=aTB + CT

c = a1bT + C

3.

Partition

$$L \rightarrow \left(\begin{array}{c|c} 1 & 0 \\ \hline l_{21} & L_{22} \end{array}\right), z \rightarrow \left(\begin{array}{c} \zeta_1 \\ \hline z_2 \end{array}\right), b \rightarrow \left(\begin{array}{c} \beta_1 \\ \hline b_2 \end{array}\right)$$

Consider

$$Lz = b$$

Substitute:

$$\underbrace{\left(\begin{array}{c|c} 1 & 0 \\ \hline l_{21} & L_{22} \end{array}\right)\left(\begin{array}{c} \zeta_1 \\ \hline z_2 \end{array}\right)}_{\left(\begin{array}{c} \zeta_1 \\ \hline l_{21}\zeta_1 + L_{22}z_2 \end{array}\right)} = \left(\begin{array}{c} \beta_1 \\ \hline b_2 \end{array}\right)$$

$$\left(\begin{array}{c} \zeta_1 = \beta_1 \\ \hline L_{22}z_2 = -l_{21}\zeta_1 + b_2 \end{array}\right)$$

Consider

$$Lz = b$$

$$\left(\begin{array}{c} \zeta_1 = \beta_1 \\ \hline L_{22}z_2 = -l_{21}\zeta_1 + b_2 \end{array}\right)$$

$$\left(\begin{array}{c} \zeta_1 := \beta_1 \\ \hline b_2 := -\zeta_1 l_{21} + b_2 \end{array}\right)$$

$L_{22}z_2 = b_2$

axpy

→ scalar로 만들기 위해서 b2를 왼쪽 변으로 빼냈다.

- 질문!!! 요게 무슨말인가?? 빼면 되는 것인가?? 무슨 말이지??? L22z2=b2. 이 식은 어디서 나온거지?? 아직도 이해를 못하겠다. ㅎㅎㅎㅎ
- 할튼, 알고리즘을 정리하면 (윗부분이 이해가 가야 다음이 진행될 것이다.

Algorithm: $[b] := \text{LTRSV\_UNB\_VAR1}(L, b)$

Partition $L \rightarrow \left(\begin{array}{c|c} L_{TL} & 0 \\ \hline L_{BL} & L_{BR} \end{array}\right), b \rightarrow \left(\begin{array}{c} b_T \\ \hline b_B \end{array}\right)$

where L_{TL} is 0×0, b_T has 0 rows

while $m(L_{TL}) < m(L)$ **do**

Repartition

$$\left(\begin{array}{c|c} L_{TL} & 0 \\ \hline L_{BL} & L_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} L_{00} & 0 & 0 \\ \hline l_{10}^T & \lambda_{11} & 0 \\ \hline L_{20} & l_{21} & L_{22} \end{array}\right), \left(\begin{array}{c} b_T \\ \hline b_B \end{array}\right) \rightarrow \left(\begin{array}{c} b_0 \\ \hline \beta_1 \\ \hline b_2 \end{array}\right)$$

$b_2 := b_2 - \beta_1 l_{21}$ axpy

Continue with

$$\left(\begin{array}{c|c} L_{TL} & 0 \\ \hline L_{BL} & L_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} L_{00} & 0 & 0 \\ \hline l_{10}^T & \lambda_{11} & 0 \\ \hline L_{20} & l_{21} & L_{22} \end{array}\right), \left(\begin{array}{c} b_T \\ \hline b_B \end{array}\right) \leftarrow \left(\begin{array}{c} b_0 \\ \hline \beta_1 \\ \hline b_2 \end{array}\right)$$

endwhile

4. 질문

Partition

$$U \rightarrow \left(\begin{array}{c|c} \upsilon_{11} & u_{12}^T \\ \hline 0 & U_{22} \end{array}\right), x \rightarrow \left(\begin{array}{c} \chi_1 \\ \hline x_2 \end{array}\right) b \rightarrow \left(\begin{array}{c} \beta_1 \\ \hline b_2 \end{array}\right)$$

Consider

$$Ux = b$$

$$\left(\begin{array}{c} \upsilon_{11}\chi_1 + u_{12}^T x_2 = \beta_1 \\ \hline U_{22}x_2 = b_2 \end{array}\right)$$

$$\left(\begin{array}{c} \chi_1 := (\beta_1 - u_{12}^T x_2)/\upsilon_{11} \\ \hline b_2 = x_2 \end{array}\right)$$

→ x2를 안다고 해보자. X2가 overwritten b2라고 해보자. 질문 : 와 왜 b2가 x2가 되는 것인가???

* 그리스 기호 모음

Matrix	Vector	Scalar			Note
		Symbol	LaTeX	Code	
A	a	α	\alpha	alpha	
B	b	β	\beta	beta	
C	c	γ	\gamma	gamma	
D	d	δ	\delta	delta	
E	e	ϵ	\epsilon	epsilon	$e_j = j$th unit basis vector.
F	f	ϕ	\phi	phi	
G	g	ξ	\xi	xi	
H	h	η	\eta	eta	
I					Used for identity matrix.
K	k	κ	\kappa	kappa	
L	l	λ	\lambda	lambda	
M	m	μ	\mu	mu	$m(\cdot)$ = row dimension.
N	n	ν	\nu	nu	ν is shared with V. $n(\cdot)$ = column dimension.
P	p	π	\pi	pi	
Q	q	θ	\theta	theta	
R	r	ρ	\rho	rho	
S	s	σ	\sigma	sigma	
T	t	τ	\tau	tau	
U	u	υ	\upsilon	upsilon	
V	v	ν	\nu	nu	ν shared with N.
W	w	ω	\omega	omega	
X	x	χ	\chi	chi	
Y	y	ψ	\psi	psi	
Z	z	ζ	\zeta	zeta	

<3단원 Summary>.

Special Matrices

Name	Represents linear transformation	Has entries
Zero matrix, $0_{m\times n} \in \mathbb{R}^{m\times n}$	$L_0 : \mathbb{R}^n \to \mathbb{R}^m$ $L_0(x) = 0$ for all x	$0 = 0_{m\times n} = \begin{pmatrix} 0 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \end{pmatrix}$
Identity matrix, $I \in \mathbb{R}^{n\times n}$	$L_I : \mathbb{R}^n \to \mathbb{R}^n$ $L_I(x) = x$ for all x	$I = I_{n\times n} = \begin{pmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{pmatrix}$
Diagonal matrix, $D \in \mathbb{R}^{n\times n}$	$L_D : \mathbb{R}^n \to \mathbb{R}^n$ if $y = L_D(x)$ then $\psi_i = \delta_i\chi_i$	$D = \begin{pmatrix} \delta_0 & 0 & \cdots & 0 \\ 0 & \delta_1 & \cdots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & \delta_{n-1} \end{pmatrix}$

Triangular matrices

$A \in \mathbb{R}^{n\times n}$ is said to be...	if ...	
lower triangular	$\alpha_{i,j} = 0$ if $i < j$	$\begin{pmatrix} \alpha_{0,0} & 0 & \cdots & 0 & 0 \\ \alpha_{1,0} & \alpha_{1,1} & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \alpha_{n-2,0} & \alpha_{n-2,1} & \cdots & \alpha_{n-2,n-2} & 0 \\ \alpha_{n-1,0} & \alpha_{n-1,1} & \cdots & \alpha_{n-1,n-2} & \alpha_{n-1,n-1} \end{pmatrix}$
strictly lower triangular	$\alpha_{i,j} = 0$ if $i \leq j$	$\begin{pmatrix} 0 & 0 & \cdots & 0 & 0 \\ \alpha_{1,0} & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \alpha_{n-2,0} & \alpha_{n-2,1} & \cdots & 0 & 0 \\ \alpha_{n-1,0} & \alpha_{n-1,1} & \cdots & \alpha_{n-1,n-2} & 0 \end{pmatrix}$
unit lower triangular	$\alpha_{i,j} = \begin{cases} 0 & \text{if } i < j \\ 1 & \text{if } i = j \end{cases}$	$\begin{pmatrix} 1 & 0 & \cdots & 0 & 0 \\ \alpha_{1,0} & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \alpha_{n-2,0} & \alpha_{n-2,1} & \cdots & 1 & 0 \\ \alpha_{n-1,0} & \alpha_{n-1,1} & \cdots & \alpha_{n-1,n-2} & 1 \end{pmatrix}$
upper triangular	$\alpha_{i,j} = 0$ if $i > j$	$\begin{pmatrix} \alpha_{0,0} & \alpha_{0,1} & \cdots & \alpha_{0,n-2} & \alpha_{0,n-1} \\ 0 & \alpha_{1,1} & \cdots & \alpha_{1,n-2} & \alpha_{1,n-1} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & \alpha_{n-2,n-2} & \alpha_{n-2,n-1} \\ 0 & 0 & \cdots & 0 & \alpha_{n-1,n-1} \end{pmatrix}$
strictly upper triangular	$\alpha_{i,j} = 0$ if $i \geq j$	$\begin{pmatrix} 0 & \alpha_{0,1} & \cdots & \alpha_{0,n-2} & \alpha_{0,n-1} \\ 0 & 0 & \cdots & \alpha_{1,n-2} & \alpha_{1,n-1} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 0 & \alpha_{n-2,n-1} \\ 0 & 0 & \cdots & 0 & 0 \end{pmatrix}$
unit upper triangular	$\alpha_{i,j} = \begin{cases} 0 & \text{if } i > j \\ 1 & \text{if } i = j \end{cases}$	$\begin{pmatrix} 1 & \alpha_{0,1} & \cdots & \alpha_{0,n-2} & \alpha_{0,n-1} \\ 0 & 1 & \cdots & \alpha_{1,n-2} & \alpha_{1,n-1} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & \alpha_{n-2,n-1} \\ 0 & 0 & \cdots & 0 & 1 \end{pmatrix}$

Transpose matrix

$$\begin{pmatrix} \alpha_{0,0} & \alpha_{0,1} & \cdots & \alpha_{0,n-2} & \alpha_{0,n-1} \\ \alpha_{1,0} & \alpha_{1,1} & \cdots & \alpha_{1,n-2} & \alpha_{1,n-1} \\ \vdots & \vdots & & \vdots & \vdots \\ \alpha_{m-2,0} & \alpha_{m-2,1} & \cdots & \alpha_{m-2,n-2} & \alpha_{m-2,n-1} \\ \alpha_{m-1,0} & \alpha_{m-1,1} & \cdots & \alpha_{m-1,n-2} & \alpha_{m-1,n-1} \end{pmatrix}^T = \begin{pmatrix} \alpha_{0,0} & \alpha_{1,0} & \cdots & \alpha_{m-2,0} & \alpha_{m-1,0} \\ \alpha_{0,1} & \alpha_{1,1} & \cdots & \alpha_{m-2,1} & \alpha_{m-1,1} \\ \vdots & \vdots & & \vdots & \vdots \\ \alpha_{0,n-2} & \alpha_{1,n-2} & \cdots & \alpha_{m-2,n-2} & \alpha_{m-1,n-2} \\ \alpha_{0,n-1} & \alpha_{1,n-1} & \cdots & \alpha_{m-2,n-1} & \alpha_{m-1,n-1} \end{pmatrix}$$

Symmetric matrix

Matrix $A \in \mathbb{R}^{n \times n}$ is symmetric if and only if $A = A^T$:

$$A = \begin{pmatrix} \alpha_{0,0} & \alpha_{0,1} & \cdots & \alpha_{0,n-2} & \alpha_{0,n-1} \\ \alpha_{1,0} & \alpha_{1,1} & \cdots & \alpha_{1,n-2} & \alpha_{1,n-1} \\ \vdots & \vdots & & \vdots & \vdots \\ \alpha_{n-2,0} & \alpha_{n-2,1} & \cdots & \alpha_{n-2,n-2} & \alpha_{n-2,n-1} \\ \alpha_{n-1,0} & \alpha_{n-1,1} & \cdots & \alpha_{n-1,n-2} & \alpha_{n-1,n-1} \end{pmatrix} = \begin{pmatrix} \alpha_{0,0} & \alpha_{1,0} & \cdots & \alpha_{n-2,0} & \alpha_{n-1,0} \\ \alpha_{0,1} & \alpha_{1,1} & \cdots & \alpha_{n-2,1} & \alpha_{n-1,1} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \alpha_{0,n-2} & \alpha_{1,n-2} & \cdots & \alpha_{n-2,n-2} & \alpha_{n-1,n-2} \\ \alpha_{0,n-1} & \alpha_{1,n-1} & \cdots & \alpha_{n-2,n-1} & \alpha_{n-1,n-1} \end{pmatrix} = A^T$$

Scaling a matrix

Let $\beta \in \mathbb{R}$ and $A \in \mathbb{R}^{m \times n}$. Then

$$\begin{aligned} \beta A &= \beta \left(\begin{array}{c|c|c|c} a_0 & a_1 & \cdots & a_{n-1} \end{array} \right) = \left(\begin{array}{c|c|c|c} \beta a_0 & \beta a_1 & \cdots & \beta a_{n-1} \end{array} \right) \\ &= \beta \begin{pmatrix} \alpha_{0,0} & \alpha_{0,1} & \cdots & \alpha_{0,n-1} \\ \alpha_{1,0} & \alpha_{1,1} & \cdots & \alpha_{1,n-1} \\ \vdots & \vdots & & \vdots \\ \alpha_{m-1,0} & \alpha_{m-1,1} & \cdots & \alpha_{m-1,n-1} \end{pmatrix} = \begin{pmatrix} \beta\alpha_{0,0} & \beta\alpha_{0,1} & \cdots & \beta\alpha_{0,n-1} \\ \beta\alpha_{1,0} & \beta\alpha_{1,1} & \cdots & \beta\alpha_{1,n-1} \\ \vdots & \vdots & & \vdots \\ \beta\alpha_{m-1,0} & \beta\alpha_{m-1,1} & \cdots & \beta\alpha_{m-1,n-1} \end{pmatrix} \end{aligned}$$

Adding matrices

Let $A, B \in \mathbb{R}^{m \times n}$. Then

$$\begin{aligned} A + B &= \left(\begin{array}{c|c|c|c} a_0 & a_1 & \cdots & a_{n-1} \end{array} \right) + \left(\begin{array}{c|c|c|c} b_0 & b_1 & \cdots & b_{n-1} \end{array} \right) = \left(\begin{array}{c|c|c|c} a_0 + b_0 & a_1 + b_1 & \cdots & a_{n-1} + b_{n-1} \end{array} \right) \\ &= \begin{pmatrix} \alpha_{0,0} & \alpha_{0,1} & \cdots & \alpha_{0,n-1} \\ \alpha_{1,0} & \alpha_{1,1} & \cdots & \alpha_{1,n-1} \\ \vdots & \vdots & & \vdots \\ \alpha_{m-1,0} & \alpha_{m-1,1} & \cdots & \alpha_{m-1,n-1} \end{pmatrix} + \begin{pmatrix} \beta_{0,0} & \beta_{0,1} & \cdots & \beta_{0,n-1} \\ \beta_{1,0} & \beta_{1,1} & \cdots & \beta_{1,n-1} \\ \vdots & \vdots & & \vdots \\ \beta_{m-1,0} & \beta_{m-1,1} & \cdots & \beta_{m-1,n-1} \end{pmatrix} \\ &= \begin{pmatrix} \alpha_{0,0} + \beta_{0,0} & \alpha_{0,1} + \beta_{0,1} & \cdots & \alpha_{0,n-1} + \beta_{0,n-1} \\ \alpha_{1,0} + \beta_{1,0} & \alpha_{1,1} + \beta_{1,1} & \cdots & \alpha_{1,n-1} + \beta_{1,n-1} \\ \vdots & \vdots & & \vdots \\ \alpha_{m-1,0} + \beta_{m-1,0} & \alpha_{m-1,1} + \beta_{m-1,1} & \cdots & \alpha_{m-1,n-1} + \beta_{m-1,n-1} \end{pmatrix} \end{aligned}$$

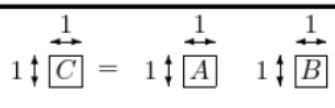
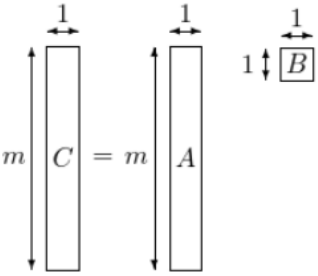
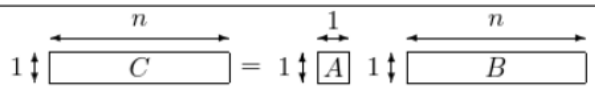
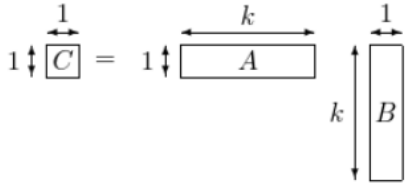
- Matrix addition commutes: $A + B = B + A$.
- Matrix addition is associative: $(A + B) + C = A + (B + C)$.
- $(A + B)^T = A^T + B^T$.

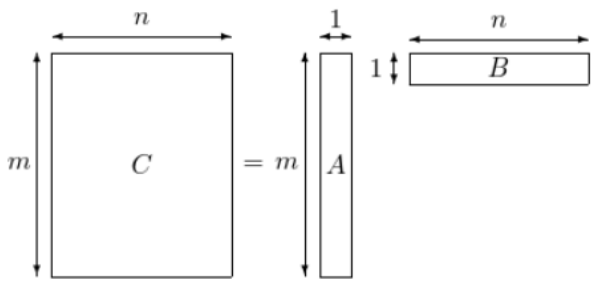
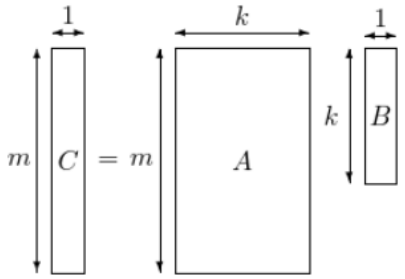
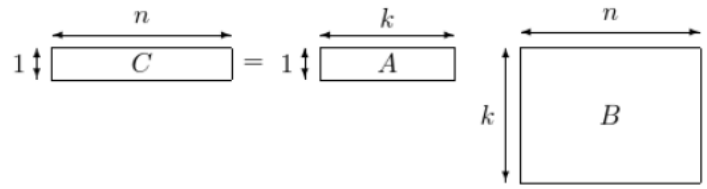
Matrix-vector multiplication

$$
\begin{aligned}
Ax &= \begin{pmatrix} \alpha_{0,0} & \alpha_{0,1} & \cdots & \alpha_{0,n-1} \\ \alpha_{1,0} & \alpha_{1,1} & \cdots & \alpha_{1,n-1} \\ \vdots & \vdots & \vdots & \vdots \\ \alpha_{m-1,0} & \alpha_{m-1,1} & \cdots & \alpha_{m-1,n-1} \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \\ \vdots \\ \chi_{n-1} \end{pmatrix} = \begin{pmatrix} \alpha_{0,0}\chi_0 + & \alpha_{0,1}\chi_1 + \cdots + & \alpha_{0,n-1}\chi_{n-1} \\ \alpha_{1,0}\chi_0 + & \alpha_{1,1}\chi_1 + \cdots + & \alpha_{1,n-1}\chi_{n-1} \\ \vdots & \vdots \quad \vdots & \vdots \\ \alpha_{m-1,0}\chi_0 + & \alpha_{m-1,1}\chi_1 + \cdots + & \alpha_{m-1,n-1}\chi_{n-1} \end{pmatrix} \\
&= \left(a_0 \middle| a_1 \middle| \cdots \middle| a_{n-1} \right) \begin{pmatrix} \chi_0 \\ \chi_1 \\ \vdots \\ \chi_{n-1} \end{pmatrix} = \chi_0 a_0 + \chi_1 a_1 + \cdots + \chi_{n-1} a_{n-1} \\
&= \begin{pmatrix} \widetilde{a}_0^T \\ \widetilde{a}_1^T \\ \vdots \\ \widetilde{a}_{m-1}^T \end{pmatrix} x = \begin{pmatrix} \widetilde{a}_0^T x \\ \widetilde{a}_1^T x \\ \vdots \\ \widetilde{a}_{m-1}^T x \end{pmatrix}
\end{aligned}
$$

<4주차 Summary>

Outer product

m	n	k	Shape	Comment
1	1	1		Scalar multiplication
m	1	1		Vector times scalar = scalar times vector
1	n	1		Scalar times row vector
1	1	k		Dot product (with row and column)

m	n	1		Outer product
m	1	k		Matrix-vector multiplication
1	n	k		Row vector times matrix multiply

5주차

Multiplication

(1) Row

(2) Column

으로 하는 방식이 다르다는 것!!

Rank-1 update

* 표기

Lowercase Greek letters (α, β, etc.) are used for scalars.

Lowercase (Roman) letters (a, b, etc) are used for vectors.

Uppercase (Roman) letters (A, B, etc) are used for matrices.

- overflow : Try to store a number larger in magnitude than this largest number, and you cause what is called an overflow. → stored as a "Not-A-Number"(NAN)

- underflow : Try to store a number not equal to zero and smaller in magnitude than this smallest number, and you cause what is called an underflow. An underflow is often set to zero.

1. 벡타

1.1. 벡타의 정의

Vectors in higher dimensions

$$x = \begin{pmatrix} \chi_0 \\ \chi_1 \\ \vdots \\ \chi_{n-1} \end{pmatrix}$$

- It is an *ordered* array.
- The entries in the array are called components.
- We start indexing the components at zero.
- The component indexed with i is denoted by χ_i.
- Each number is a real number: $\chi_i \in \mathbb{R}$. (It could be a complex number, but we will not discuss that now.)
- $x \in \mathbb{R}^n$.
- A vector has a direction and a length.
  - Draw an arrow from the origin to the point $(\chi_0, \chi_1, \ldots, \chi_{n-1})$.
  - The *length* is $\sqrt{\chi_0^2 + \chi_1^2 + \cdots + \chi_{n-1}^2}$.
- A vector does *not* have a location.

2. 표준단위 벡타 : Standard Basis Vectors (Unit basis vectors)

- 단위벡타는 크기가 1인 벡터이다. 표준단위벡터랑은 다르다.

- Rn에서는 n개의 단위 벡타가 있을 것이다.

- 축에 해당한다.

$\{e_0, e_1, \ldots, e_{n-1}\} \subset \mathbb{R}^n$ given by

$$e_0 = \begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \\ 0 \end{pmatrix}, \quad e_1 = \begin{pmatrix} 0 \\ 1 \\ \vdots \\ 0 \\ 0 \end{pmatrix}, \quad \cdots, \quad e_{n-1} = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \end{pmatrix}.$$

- The symbols $\vec{\imath}$ and $\vec{\jmath}$ denote e_0, e_1, and e_2 in 2D and 3D; and
- The symbol $\vec{k}$ denotes e_2 in $3D$.

Thus

$$\vec{\imath} = e_0 = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \quad \text{and} \quad \vec{\jmath} = e_1 = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

or

$$\vec{\imath} = e_0 \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}, \quad \vec{\jmath} = e_1 = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}, \quad \text{and} \quad \vec{k} = e_2 = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$$

1.3. 벡타의 연산(Operators)

- 개괄

Vector scaling	$\alpha x = \begin{pmatrix} \alpha\chi_0 \\ \alpha\chi_1 \\ \vdots \\ \alpha\chi_{n-1} \end{pmatrix}$
Vector addition	$x + y = \begin{pmatrix} \chi_0 + \psi_0 \\ \chi_1 + \psi_1 \\ \vdots \\ \chi_{n-1} + \psi_{n-1} \end{pmatrix}$
Vector subtraction	$x - y = \begin{pmatrix} \chi_0 - \psi_0 \\ \chi_1 - \psi_1 \\ \vdots \\ \chi_{n-1} - \psi_{n-1} \end{pmatrix}$
AXPY	$\alpha x + y = \begin{pmatrix} \alpha\chi_0 + \psi_0 \\ \alpha\chi_1 + \psi_1 \\ \vdots \\ \alpha\chi_{n-1} + \psi_{n-1} \end{pmatrix}$
dot (inner) product	$x^T y = \sum_{i=0}^{n-1} \chi_i \psi_i$
vector length	$\|x\|_2 = \sqrt{x^T x} = \sqrt{\sum_{i=0}^{n-1} \chi_i \chi_i}$

- 성질 요약

Vector Addition

- Is commutative. That is, for all vectors $x, y \in \mathbb{R}^n$, $x + y = y + x$.
- Is associative. That is, for all vectors$x, y, z \in \mathbb{R}^n$, $(x + y) + z = x + (y + z)$.
- Has the zero vector as an identity. For all vectors $x \in \mathbb{R}^n$, $x + \mathbf{0} = \mathbf{0} + x = x$ where $\mathbf{0}$ is the vector of size n with 0 for each component.
- Has an inverse, $-x$. That is $x + (-x) = \mathbf{0}$.

The dot product of vectors

- Is commutative. That is, for all vectors $x, y \in R^n$, $x^T y = y^T x$.
- Distributes over vector addition. That is, for all vectors $x, y, z \in R^n$, $x^T(y + z) = x^T y + x^T z$.Also, $(x + y)^T z = x^T z + y^T z$.

Other Properties

- For $x, y \in R^n$, $(x + y)^T(x + y) = x^T x + 2x^T y + y^T y$.
- For $x, y \in R^n$, $x^T y = 0$ if and only if x and y are orthogonal.
- Let $x, y \in R^n$ be nonzero vectors and let the angle between them equal θ. Then $cos(\theta) = \frac{x^T y}{||x||_2 ||y||_2}$.
- For $x \in R^n$, $x^T e_i = e_i^T x = \chi_i$ where χ_i equals the ith component of x.

- computation

Summary of the Routines for Vector Operations

Operation Abbrev.	Definition	Function	Approx. cost	
			flops	memops
Vector-vector operations				
Copy (COPY)	$y := x$	`laff.copy( x, y )`	0	$2n$
Vector scaling (SCAL)	$x := \alpha x$	`laff.scal( alpha, x )`	n	$2n$
Scaled addition (AXPY)	$y := \alpha x + y$	`laff.axpy( alpha, x, y )`	$2n$	$3n$
Dot product (DOT)	$\alpha := x^T y$	`alpha = laff.dot( x, y )`	$2n$	$2n$
Length (NORM2)	$\alpha := \|x\|_2$	`alpha = laff.norm2( x )`	$2n$	n

1.3.1. equal

Equality

Let $x \in \mathbb{R}^n$ and $y \in \mathbb{R}^n$ with

$$x = \begin{pmatrix} \chi_0 \\ \chi_1 \\ \vdots \\ \chi_{n-1} \end{pmatrix} \quad \text{and} \quad y = \begin{pmatrix} \psi_0 \\ \psi_1 \\ \vdots \\ \psi_{n-1} \end{pmatrix}$$

Then $x = y$ (x equals y) if and only if (iff)

$$\begin{aligned} \chi_0 &= \psi_0; && \text{and} \\ \chi_1 &= \psi_1; && \text{and} \\ &\vdots \\ \chi_{n-1} &= \psi_{n-1}. \end{aligned}$$

(Corresponding elements of x and y are equal.)

Assignment

Let $x \in \mathbb{R}^n$ and $y \in \mathbb{R}^n$ with

$$x = \begin{pmatrix} \chi_0 \\ \chi_1 \\ \vdots \\ \chi_{n-1} \end{pmatrix} \quad \text{and} \quad y = \begin{pmatrix} \psi_0 \\ \psi_1 \\ \vdots \\ \psi_{n-1} \end{pmatrix}.$$

Then

$y := x$ (read: y becomes x)

assigns components of x to the corresponding components of y:

$$\begin{aligned} \psi_0 &:= \chi_0; && \text{and} \\ \psi_1 &:= \chi_1; && \text{and} \\ &\vdots \\ \psi_{n-1} &:= \chi_{n-1}. \end{aligned}$$

→ computer에서는 :=를 쓴다. Quality를 같다고 하는 것과는 다르다.

$y := x$:

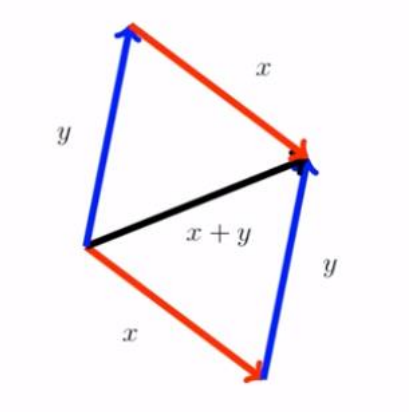
```
for i = 0, ..., n − 1
    ψi := χi
endfor
```

→ for loop을 사용하면 된다.

1.3.2. 덧셈

Let $x \in \mathbb{R}^n$ and $y \in \mathbb{R}^n$ Then

$$x + y = \begin{pmatrix} \chi_0 \\ \chi_1 \\ \vdots \\ \chi_{n-1} \end{pmatrix} + \begin{pmatrix} \psi_0 \\ \psi_1 \\ \vdots \\ \psi_{n-1} \end{pmatrix} = \begin{pmatrix} \chi_0 + \psi_0 \\ \chi_1 + \psi_1 \\ \vdots \\ \chi_{n-1} + \psi_{n-1} \end{pmatrix}$$



- computing 방법

Algorithm

Computing $z := x + y$:

$$\underbrace{\begin{pmatrix} \zeta_0 \\ \zeta_1 \\ \vdots \\ \zeta_{n-1} \end{pmatrix}}_{z} := \underbrace{\begin{pmatrix} \chi_0 \\ \chi_1 \\ \vdots \\ \chi_{n-1} \end{pmatrix}}_{x} + \underbrace{\begin{pmatrix} \psi_0 \\ \psi_1 \\ \vdots \\ \psi_{n-1} \end{pmatrix}}_{y} = \underbrace{\begin{pmatrix} \chi_0 + \psi_0 \\ \chi_1 + \psi_1 \\ \vdots \\ \chi_{n-1} + \psi_{n-1} \end{pmatrix}}_{x+y}.$$

for $i = 0, \ldots, n-1$
  $\zeta_i := \chi_i + \psi_i$
endfor

1.3.3. 상수곱(scaling)

Computing $y := \alpha x$:

$$\underbrace{\begin{pmatrix} \psi_0 \\ \psi_1 \\ \vdots \\ \psi_{n-1} \end{pmatrix}}_{y} := \alpha \underbrace{\begin{pmatrix} \chi_0 \\ \chi_1 \\ \vdots \\ \chi_{n-1} \end{pmatrix}}_{x} = \underbrace{\begin{pmatrix} \alpha\chi_0 \\ \alpha\chi_1 \\ \vdots \\ \alpha\chi_{n-1} \end{pmatrix}}_{\alpha x}.$$

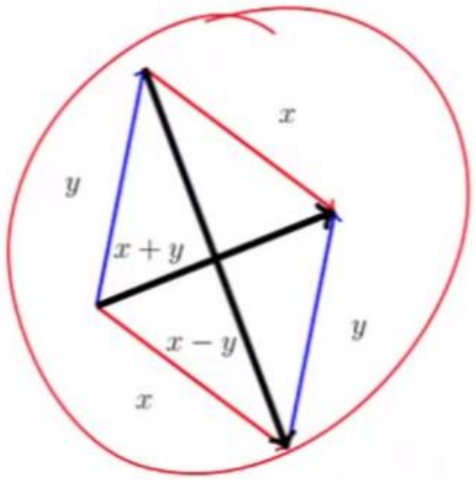
for $i = 0, \ldots, n-1$
  $\psi_i := \alpha\chi_i$
endfor

```
if ~isvector( x )
    x_out = 'FAILED';
    return
end

if ( n_x == 1 )     % x is a column vector
    for i=1:m_x
        x( i,1 ) = alpha * x( i,1 );
    end
else    % x is a row vector
    for i=1:n_x
        x( 1,i ) = alpha * x( 1,i );
    end
end
```

1.3.4. 뺄셈 : 덧셈과 scaling을 동시에 하는 것이다.

Summary



$$x - y = \begin{pmatrix} \chi_0 \\ \chi_1 \\ \vdots \\ \chi_{n-1} \end{pmatrix} - \begin{pmatrix} \psi_0 \\ \psi_1 \\ \vdots \\ \psi_{n-1} \end{pmatrix} = \begin{pmatrix} \chi_0 - \psi_0 \\ \chi_1 - \psi_1 \\ \vdots \\ \chi_{n-1} - \psi_{n-1} \end{pmatrix}.$$

1.3.5. 내적(Dot, inner product)

(승연님 질문) Dot과 axpy의 차이점은 무엇인가? : 내적은 하나의 스칼라값으로 나오는 것이고, axpy는 linear combination으로 가는 다리이다.

- 정의 :

Let $x, y \in \mathbb{R}^n$. Let

$$x = \begin{pmatrix} \chi_0 \\ \chi_1 \\ \vdots \\ \chi_{n-1} \end{pmatrix} \quad \text{and} \quad y = \begin{pmatrix} \psi_0 \\ \psi_1 \\ \vdots \\ \psi_{n-1} \end{pmatrix}.$$

Then the dot product (also called the inner product) of x and y is given by

$$\text{dot}(x, y) = \chi_0\psi_0 + \chi_1\psi_1 + \cdots + \chi_{n-1}\psi_{n-1} = \sum_{i=0}^{n-1} \chi_i\psi_i.$$

- 다른 표현 : transposition을 해서 곱하면 된다. x^Ty

Important: Alternative Notation

We will often write

$$\begin{aligned} \text{dot}(x, y) = x^T y &= \begin{pmatrix} \chi_0 \\ \chi_1 \\ \vdots \\ \chi_{n-1} \end{pmatrix}^T \begin{pmatrix} \psi_0 \\ \psi_1 \\ \vdots \\ \psi_{n-1} \end{pmatrix} \\ &= \begin{pmatrix} \chi_0 & \chi_1 & \cdots & \chi_{n-1} \end{pmatrix} \begin{pmatrix} \psi_0 \\ \psi_1 \\ \vdots \\ \psi_{n-1} \end{pmatrix} \\ &= \chi_0\psi_0 + \chi_1\psi_1 + \cdots + \chi_{n-1}\psi_{n-1}. \end{aligned}$$

- 의미 : 내적이 0이면 수직이다.

1.3.5.1. computing method

Algorithm

$$\alpha := \text{dot}\left(\begin{pmatrix} \chi_0 \\ \chi_1 \\ \vdots \\ \chi_{n-1} \end{pmatrix}, \begin{pmatrix} \psi_0 \\ \psi_1 \\ \vdots \\ \psi_{n-1} \end{pmatrix}\right) = \chi_0\psi_0 + \chi_1\psi_1 + \cdots + \chi_{n-1}\psi_{n-1}.$$

Alternatively:

$$\alpha := \chi_{n-1}\psi_{n-1}. + (\cdots + (\chi_1\psi_1 + (\chi_0\psi_0 + 0))\cdots).$$

$\alpha := 0$
for $i = 0, \ldots, n-1$
  $\alpha := \chi_i\psi_i + \alpha$
endfor

```
if ( n_x == 1 )     % x is a column vector
    if ( n_y == 1 )     % y is a column vector
        % Copy the elements of x into the elements of y
        for i=1:m_x
            alpha = alpha + y( i,1 ) * x( i,1 );
        end
    else     % y is a row vector
        % Copy the elements of x into the elements of y
        for i=1:m_x
            y( 1,i ) = alpha + x( i,1 ) * y( 1,i );
        end
    end
else    % x is a row vector
```

→ alpha = 0으로 하고 계속 더해준다.

1.3.5.2. Slicing and dicing

- 하는 이유 : helps express algorithms without worrying about indices(index)

$x^T y$

$$\left\{ \begin{pmatrix} 2 \\ -1 \end{pmatrix}^T \begin{pmatrix} 1 \\ -2 \end{pmatrix} \right\} \left\{ \begin{pmatrix} 4 \\ 2 \\ 1 \end{pmatrix} \begin{pmatrix} 2 \\ 3 \\ -1 \end{pmatrix} \right\}$$

$$= \underbrace{(2) \times (1) + (-1) \times (-2)}_{\begin{pmatrix} 2 \\ -1 \end{pmatrix}^T \begin{pmatrix} 1 \\ -2 \end{pmatrix}} + \underbrace{(4) \times (2) + (2) \times (3) + (1) \times (-1)}_{\begin{pmatrix} 4 \\ 2 \\ 1 \end{pmatrix}^T \begin{pmatrix} 2 \\ 3 \\ -1 \end{pmatrix}}$$

$$x^T y = \begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \\ \chi_3 \\ \chi_4 \end{pmatrix}^T \begin{pmatrix} \psi_0 \\ \psi_1 \\ \psi_2 \\ \psi_3 \\ \psi_4 \end{pmatrix} = \sum_{i=0}^{n-1} \chi_i \psi_i \qquad n=5$$

$$= (\chi_0 \times \psi_0 + \chi_1 \times \psi_1 + \chi_2 \times \psi_2) + (\chi_3 \times \psi_3 + \chi_4 \times \psi_4)$$

$$= \sum_{i=0}^{2} \chi_i \times \psi_i + \sum_{i=3}^{4} \chi_i \times \psi_i$$

$$x^T y = \begin{pmatrix} x_0 \\ x_1 \end{pmatrix}^T \begin{pmatrix} y_0 \\ y_1 \end{pmatrix}$$

$$= x_0^T y_0 + x_1^T y_1$$

→ sub-vector가 same size인 것이 중요하다.

An important special case

$$\left(\begin{array}{r} 2 \\ \hline -1 \\ \hline 4 \\ \hline 2 \\ 1 \end{array}\right)^T \left(\begin{array}{r} 1 \\ \hline -2 \\ \hline 2 \\ \hline 3 \\ -1 \end{array}\right) = \begin{pmatrix} 2 \\ -1 \end{pmatrix}^T \begin{pmatrix} 1 \\ -2 \end{pmatrix} + 4 \times 2 + \begin{pmatrix} 2 \\ 1 \end{pmatrix}^T \begin{pmatrix} 3 \\ -1 \end{pmatrix}$$

$$x^T y = \left(\begin{array}{c} x_0 \\ \hline \chi_1 \\ \hline x_2 \end{array}\right)^T \left(\begin{array}{c} y_0 \\ \hline \psi_1 \\ \hline y_2 \end{array}\right) = x_0^T y_0 + \chi_1 \times \psi_1 + x_2^T y_2$$

→ 가운데가 scalar일 수도 있구나

- 내적의 slicing algorithm

Algorithm: $[\alpha] := \text{DOT}(x, y)$

Partition $x \rightarrow \left(\begin{array}{c} x_T \\ \hline x_B \end{array}\right)$, $y \rightarrow \left(\begin{array}{c} y_T \\ \hline y_B \end{array}\right)$

where x_T and y_T have 0 elements

$\alpha := 0$

while $m(x_T) < m(x)$ **do**

Repartition

$$\left(\begin{array}{c} x_T \\ \hline x_B \end{array}\right) \rightarrow \left(\begin{array}{c} x_0 \\ \hline \chi_1 \\ \hline x_2 \end{array}\right), \left(\begin{array}{c} y_T \\ \hline y_B \end{array}\right) \rightarrow \left(\begin{array}{c} y_0 \\ \hline \psi_1 \\ \hline y_2 \end{array}\right)$$

$\alpha := \chi_1 \times \psi_1 + \alpha$

Continue with

$$\left(\begin{array}{c} x_T \\ \hline x_B \end{array}\right) \leftarrow \left(\begin{array}{c} x_0 \\ \hline \chi_1 \\ \hline x_2 \end{array}\right), \left(\begin{array}{c} y_T \\ \hline y_B \end{array}\right) \leftarrow \left(\begin{array}{c} y_0 \\ \hline \psi_1 \\ \hline y_2 \end{array}\right)$$

endwhile

$$\left(\begin{array}{r} 2 \\ \hline -1 \\ \hline 4 \\ \hline 2 \\ 1 \end{array}\right)^T \left(\begin{array}{r} 1 \\ \hline -2 \\ \hline 2 \\ \hline 3 \\ -1 \end{array}\right)$$

$$\begin{aligned} &\alpha = 0 \\ &+(2) \times (1) \\ &+(-1) \times (-2) \\ &+(4) \times (2) \\ &+(2) \times (3) \\ &+(1) \times (-1) \end{aligned}$$

20 / 20

→ 한칸씩 한칸씩 전진한다.

Algorithm: $[\alpha] := \text{DOT}(x, y)$

Partition $x \rightarrow \left(\frac{x_T}{x_B}\right)$, $y \rightarrow \left(\frac{y_T}{y_B}\right)$

where x_T and y_T have 0 elements

$\alpha := 0$

while $m(x_T) < m(x)$ **do**

Repartition

$$\left(\frac{x_T}{x_B}\right) \rightarrow \left(\frac{x_0}{\chi_1}{x_2}\right), \left(\frac{y_T}{y_B}\right) \rightarrow \left(\frac{y_0}{\psi_1}{y_2}\right)$$

where χ_1 has 1 row, ψ_1 has 1 row

$\alpha := \chi_1 \times \psi_1 + \alpha$

Continue with

$$\left(\frac{x_T}{x_B}\right) \leftarrow \left(\frac{x_0}{\chi_1}{x_2}\right), \left(\frac{y_T}{y_B}\right) \leftarrow \left(\frac{y_0}{\psi_1}{y_2}\right)$$

endwhile

Most Visited Piazza hourly Facebook 3PLAY FTP Spark Canvas

```
function [ alpha_out ] = Dot_unb( alpha, x, y )

  [ xT, ...
    xB ] = FLA_Part_2x1( x, ...
                         0, 'FLA_TOP' );

  [ yT, ...
    yB ] = FLA_Part_2x1( y, ...
                         0, 'FLA_TOP' );

  while ( size( xT, 1 ) < size( x, 1 ) )

    [ x0, ...
      chi1, ...
      x2 ] = FLA_Repart_2x1_to_3x1( xT, ...
                                    xB, ...
                                    1, 'FLA_BOTTOM' );

    [ y0, ...
      psi1, ...
      y2 ] = FLA_Repart_2x1_to_3x1( yT, ...
                                    yB, ...
                                    1, 'FLA_BOTTOM' );

    %-------------------------------------------------------------%

    %                       update line 1                         %
    %                             :                               %
    %                       update line n                         %

    %-------------------------------------------------------------%

    [ xT, ...
      xB ] = FLA_Cont_with_3x1_to_2x1( x0, ...
                                       chi1, ...
                                       x2, ...
                                       'FLA_TOP' );

    [ yT, ...
      yB ] = FLA_Cont_with_3x1_to_2x1( y0, ...
                                       psi1, ...
```

1.4. Axpy operation : a multiply plus y

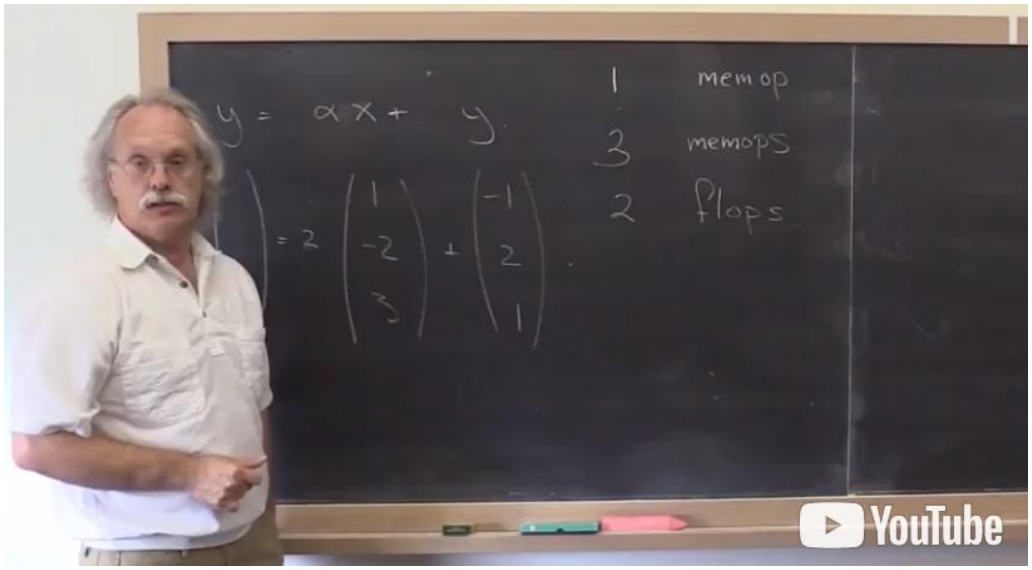
$$y := \alpha x + y = \begin{pmatrix} \alpha\chi_0 + \psi_0 \\ \alpha\chi_1 + \psi_1 \\ \vdots \\ \alpha\chi_{n-1} + \psi_{n-1} \end{pmatrix}.$$

for $i = 0, \ldots, n-1$

$\psi_i := \alpha\chi_i + \psi_i$

endfor

- axpy를 하는 데에 필요한 operation의 수 세기 (Cost)



(1) memops(memory operations)

- 일단 2(a에 해당)를 memory에 넣어서 자주 꺼내 쓸 수 있게 할 수 있다.
- 1과 -1을 읽는데 2개가 사용되고, 그것을 연산해서 writing하는데 1개 → 3개

→ 3n+1

(2) floating point operations

- floating point operation이란 연산을 얼마나 하냐 인 것 같다.
- 곱하고 더하고 2번 이므로 2이다.

➔ 2n

1.5. linear combination ← 아까의 axpy의 확장판이라고도 볼 수 있다.

Let $u, v \in \mathbb{R}^m$ and $\alpha, \beta \in \mathbb{R}$. The following is called a linear combination of the vectors u and v with coefficients α and β:

$$\alpha u + \beta v = \alpha \begin{pmatrix} v_0 \\ v_1 \\ \vdots \\ v_{m-1} \end{pmatrix} + \beta \begin{pmatrix} \nu_0 \\ \nu_1 \\ \vdots \\ \nu_{m-1} \end{pmatrix}$$

$$= \begin{pmatrix} \alpha v_0 \\ \alpha v_1 \\ \vdots \\ \alpha v_{m-1} \end{pmatrix} + \begin{pmatrix} \beta \nu_0 \\ \beta \nu_1 \\ \vdots \\ \beta \nu_{m-1} \end{pmatrix} = \begin{pmatrix} \alpha v_0 + \beta \nu_0 \\ \alpha v_1 + \beta \nu_1 \\ \vdots \\ \alpha v_{m-1} + \beta \nu_{m-1} \end{pmatrix}$$

If $v_0, \ldots, v_{n-1} \in \mathbb{R}^m$ and $\chi_0, \ldots, \chi_{n-1} \in \mathbb{R}$, then

$$\chi_0 v_0 + \chi_1 v_1 + \cdots + \chi_{n-1} v_{n-1} = \sum_{j=0}^{n-1} \chi_j v_j.$$

is a linear combination of the vectors, with coefficients $\chi_0, \ldots, \chi_{n-1}$.

1.5.1. Computing method : Axpy를 여러번 하면 된다.

Algorithm

$$w := \chi_0 v_0 + \chi_1 v_1 + \cdots + \chi_{n-1} v_{n-1}.$$

Alternatively:

$$w := \chi_{n-1} v_{n-1} + (\cdots + (\chi_1 v_1 + (\chi_0 v_0 + 0))) \cdots).$$

$w = 0$
for $j = 0, \ldots, n-1$
  $w := \chi_j v_j + w$
endfor

```
    for i=1:n_x
        y( i,1 ) = alpha * x( 1,i ) + y( i,1 );
    end
else     % y is a row vector
    % Copy the elements of x into the elements of y
    for i=1:n_x
        y( 1,i ) = alpha * x( 1,i ) + y( i,1 );
    end
end
```

→ axpy를 계속 해나가면 된다. 일단 0부터 시작해서 1개씩 하면 된다. 그러면 w는 linear combination이 될 것이다.

- 예를 들면, 단위벡타의 linear combination이 일반적인 vector가 된다.

Example: linear combination of unit basis vector

Let $x \in \mathbb{R}^n$

$$x = \begin{pmatrix} \chi_0 \\ \chi_1 \\ \vdots \\ \chi_{n-1} \end{pmatrix}.$$

Then

$$x = \begin{pmatrix} \chi_0 \\ \chi_1 \\ \vdots \\ \chi_{n-1} \end{pmatrix} = \chi_0 \begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \\ 0 \end{pmatrix} + \chi_1 \begin{pmatrix} 0 \\ 1 \\ \vdots \\ 0 \\ 0 \end{pmatrix} + \cdots + \chi_{n-1} \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \end{pmatrix}$$

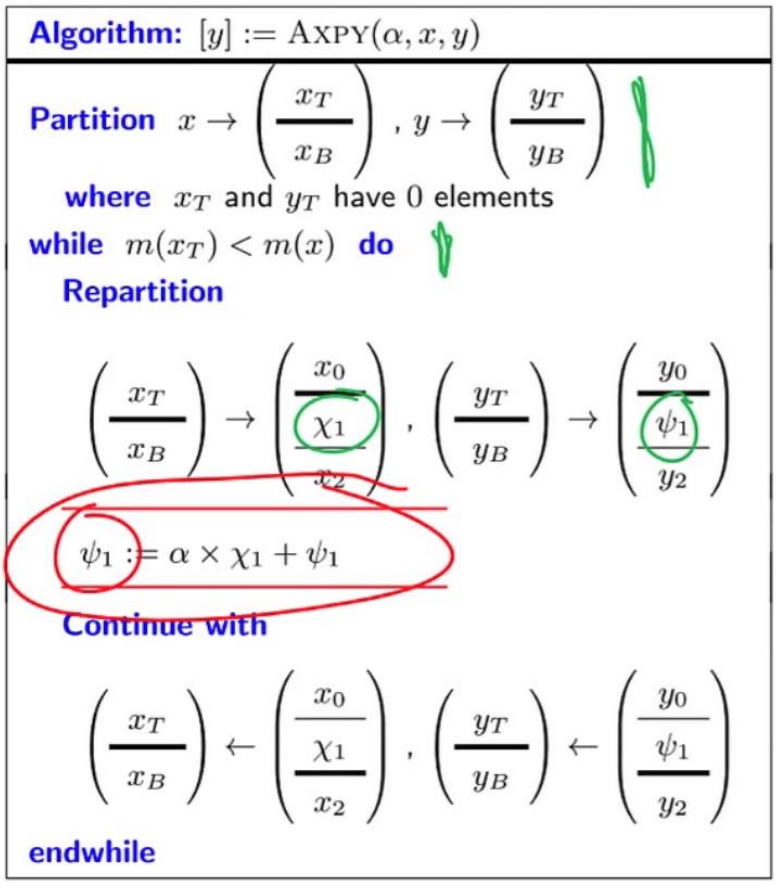
$$= \chi_0 e_0 + \chi_1 e_1 + \cdots + \chi_{n-1} e_{n-1} = \sum_{i=0}^{n-1} \chi_i e_i.$$

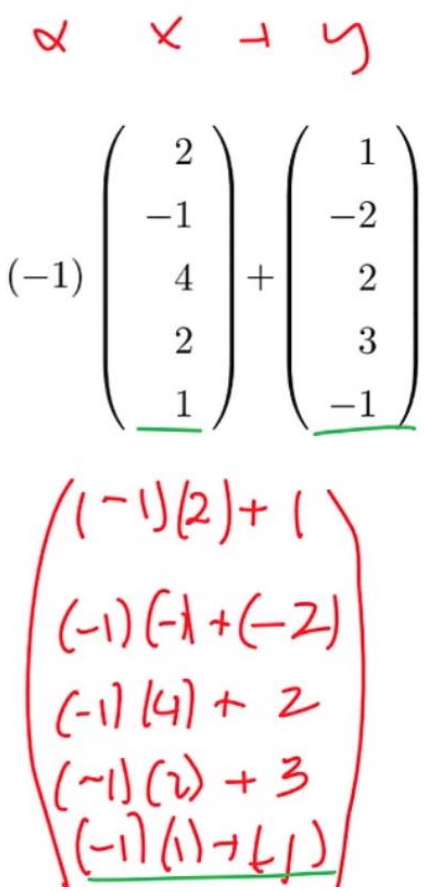
1.5.2. 계산의 다른 방식 : slicing and dicing

More generally

$$\alpha x + y = \alpha \begin{pmatrix} x_0 \\ x_1 \end{pmatrix} + \begin{pmatrix} y_0 \\ y_1 \end{pmatrix} = \begin{pmatrix} \alpha x_0 + y_0 \\ \alpha x_1 + y_1 \end{pmatrix}$$

$$\alpha x + y = \alpha \begin{pmatrix} x_0 \\ \hline x_1 \\ \hline \vdots \\ \hline x_{N-1} \end{pmatrix} + \begin{pmatrix} y_0 \\ \hline y_1 \\ \hline \vdots \\ \hline y_{N-1} \end{pmatrix} = \begin{pmatrix} \alpha x_0 + y_0 \\ \hline \alpha x_1 + y_1 \\ \hline \vdots \\ \hline \alpha x_{N-1} + y_{N-1} \end{pmatrix}$$





1.6. Vector length cf) size랑 다르다 → component의 개수)

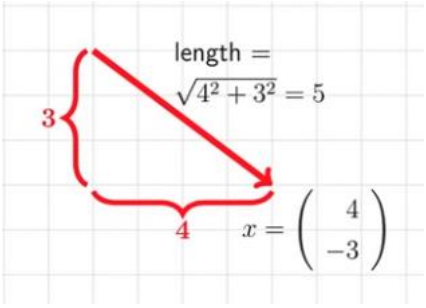
- Norm 함수 : A norm is a function, in our case of a vector in R^n, that maps every vector to a nonnegative real number. One example is the (Euclidean) length of a vector, which we call the 2-norm: for x∈Rn,

Cf) 다른 norm 함수 :

- taxi-cab norm
  - For $1 \leq p \leq \infty$, the p-norm:

$$\|x\|_p = \sqrt[p]{\sum_{i=0}^{n-1} |\chi_i|^p} = \left(\sum_{i=0}^{n-1} |\chi_i|^p\right)^{1/p}.$$

- 기하학적 의미 :



- 식 :

Length of a vector of arbitrary size

Let $x \in \mathbb{R}^n$ and $x = \begin{pmatrix} \chi_0 \\ \chi_1 \\ \vdots \\ \chi_{n-1} \end{pmatrix}$.

The (Euclidean) length of x is given by

$$\sqrt{\chi_0^2 + \chi_1^2 + \cdots + \chi_{n-1}^2} = \sqrt{\sum_{i=0}^{n-1} \chi_i^2}.$$

1.6.1. computing method

x^Tx내적의 제곱근($\sqrt{dot(x,x)}$)이 length(x)이다!! : The Euclidean length of a vector equals the square root of the sum of the squares of its component

Observation

Let $x \in \mathbb{R}^n$. Then

$$\text{dot}(x,x) = x^T x = \begin{pmatrix} \chi_0 \\ \chi_1 \\ \vdots \\ \chi_{n-1} \end{pmatrix}^T \begin{pmatrix} \chi_0 \\ \chi_1 \\ \vdots \\ \chi_{n-1} \end{pmatrix}$$

$$= \begin{pmatrix} \chi_0 & \chi_1 & \cdots & \chi_{n-1} \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \\ \vdots \\ \chi_{n-1} \end{pmatrix}$$

$$= \chi_0^2 + \chi_1^2 + \cdots + \chi_{n-1}^2 = \sum_{i=0}^{n-1} \chi_i^2.$$

laff_norm2.m

```
function [ alpha ] = laff_norm2( x )
%UNTITLED2 Summary of this function goes here
%   Detailed explanation goes here

alpha = sqrt( laff_dot( x, x ) );

end

```

1.7. Vector function

- 정의 : function of one of more scalars and/or vectors whose output is a vector. 함수의 산출이 vector이기만 한면 된다. 근데 scalar로 size가 1인 vector이므로 산출물이 scalar여도 된다.

- 예시 :

$$f(\alpha,\beta) = \begin{pmatrix} \alpha+\beta \\ \alpha-\beta \end{pmatrix}$$

스칼라 2개로도 만들 수 있다. (스칼라는 그릭 소문자로)

$$f(\alpha)\left(\begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix}\right) = \begin{pmatrix} \chi_0+\alpha \\ \chi_1+\alpha \\ \chi_2+\alpha \end{pmatrix}$$

The AXPY and DOT vector functions are other functions that we have already encountered.

$$axpy(\alpha, x, y) = \alpha x + y$$

- linear transformation을 배우기 위한 초석이다.

- 즉, vector functions to **map a vector to vector**. → linear transformation이 바로 vector function의 대표다.

1.8. 행렬의 computing method

Vector-vector operations			
Copy (*copy*)	$y := x$	`y = laff_copy( x, y )`	`y = x`
Vector scaling (*scal*)	$x := \alpha x$	`x = laff_scal( alpha, x )`	`x = alpha * x`
Scaled addition (*axpy*)	$y := \alpha x + y$	`y = laff_axpy( alpha, x, y )`	`y = alpha x + y`
Dot product (*dot*)	$\alpha := x^T y$	`alpha = laff_dot( x, y )`	`alpha = x' * y`
Length (*norm2*)	$\alpha := \|x\|_2$	`alpha = laff_norm2( x )`	`alpha = norm2( x )`

- 이렇게 하면 됨

- ; 를 하면 프린팅이 안 됨

In M-script (MATLAB), this can be entered as

```
A = [
1 2 3
4 5 6
7 8 9
]
```

Now, a (column) vector can be entered as

```
x = [
1
4
7
]
```

and is hence stored just like a matrix with only one column.

```
x = [ 4 5 6 ]
```

- indexing

```
>> x(1)

ans =

     1
```

- size 보는 방법

```
>> size (x)

ans =

     3     1
```

```
% Extract the row and column sizes of x and y
[ m_x, n_x ] = size( x );
[ m_y, n_y ] = size( y );
```

- equal 개념

```
>> for i=1:3
y (i)=x(i)
end
```

laff_copy_mine.m

```m
function [ y_out ] = laff_copy( x, y )
%UNTITLED 이 함수의 요약 설명 위치
%   자세한 설명 위치
y_out = zeros(3, 1);

for i=1:3
    y_out (i)= x(i);
end
return
```

laff_copy_mine.m

```m
function [ y_out ] = laff_copy( x, y )
%UNTITLED 이 함수의 요약 설명 위치
%   자세한 설명 위치
[m_x, n_x]=size(x);
y_out = zeros(m_x, 1);

for i=1:m_x
    y_out (i)= x(i);
end
return
```

```m
% Extract the row and column sizes of x and y
[ m_x, n_x ] = size( x );
[ m_y, n_y ] = size( y );

% Make sure x and y are (row or column) vectors of equal length
if ( m_x ~= 1 && n_x ~= 1 ) | ( m_y ~= 1 && n_y ~= 1 )
    y_out = 'FAILED';
    return
end
if ( m_x * n_x ~= m_y * n_y )
    y_out = 'FAILED';
    return
end
```

```m
if ( n_x == 1 )     % x is a column vector
    if ( n_y == 1 )     % y is a column vector
        % Copy the elements of x into the elements of y
        for i=1:m_x
            y( i,1 ) = x( i,1 );
        end
    else     % y is a row vector
        % Copy the elements of x into the elements of y
        for i=1:m_x
            y( 1,i ) = x( i,1 );
        end
    end
else    % x is a row vector
```

2. Linear transformation and indices

2.1. Linear transformation의 정의와 조건

2.1.1. 정의 : 선형성을 갖는 벡타 함수, 함수인데 벡타를 대상으로 하는 vector function이다.

(1) scale, transform / add, transform 순서 상관없다.

A vector function $L : \mathbb{R}^n \to \mathbb{R}^m$ is said to be a *linear transformation* if for all $x, y \in \mathbb{R}^n$ and $\alpha \in \mathbb{R}$

- Transforming a scaled vector is the same as scaling the transformed vector:

$$L(\alpha x) = \alpha L(x)$$

- Transforming the sum of two vectors is the same as summing the two transformed vectors:

$$L(x + y) = L(x) + L(y)$$

(2) 중요한 두번째 정의

$L : \mathbb{R}^n \to \mathbb{R}^m$ is a linear transformation if and only if (iff) for all $u, v \in \mathbb{R}^n$ and $\alpha, \beta \in \mathbb{R}$

$$L(\alpha u + \beta v) = \alpha L(u) + \beta L(v).$$

Pf) 증명

($\Rightarrow$) Assume that $L : R^n \to R^m$ is a linear transformation and let $u, v \in \mathbb{R}^n$ be *arbitrary* vectors and $\alpha, \beta \in \mathbb{R}$ be *arbitrary* scalars.

Then

$$\begin{aligned} & L(\alpha u + \beta v) \\ = & \quad < \text{since } \alpha u \text{ and } \beta v \text{ are vectors and } L \text{ is a linear transformation} \\ & L(\alpha u) + L(\beta v) \\ = & \quad < \text{since } L \text{ is a linear transformation} > \\ & \alpha L(u) + \beta L(v) \end{aligned}$$

($\Leftarrow$) Assume that all $u, v \in \mathbb{R}^n$ and all $\alpha, \beta \in \mathbb{R}$ it is the case that $L(\alpha u + \beta v) = \alpha L(u) + \beta L(v)$.

We need to show that

- $L(\alpha u) = \alpha L(u)$.

  This follows immediately by setting $\beta = 0$.

- $L(u + v) = L(u) + L(v)$.

  This follows immediately by setting $\alpha = \beta = 1$.

→ 반대방향으로도 증명해야 한다. 이 식이 성립하면 linear transformation이다!

2.1.2. Linear transformation인지 판정

How to check if a vector function is a linear transformation:

- Check if $f(0) = 0$. If it isn't, it is **not** a linear transformation.
- If $f(0) = 0$ then *either*:
  - Prove it is or isn't a linear transformation from the definition:
    - Find an example where $f(\alpha x) \neq \alpha f(x)$ or $f(x + y) \neq f(x) + f(y)$. In this case the function is *not* a linear transformation; or
    - Prove that $f(\alpha x) = \alpha f(x)$ and $f(x + y) = f(x) + f(y)$ for all α, x, y.

    or

  - Compute the *possible* matrix A that represents it and see if $f(x) = Ax$. If it is equal, it is a linear transformation. If it is not, it is not a linear transformation.

(1) 공식 조건 2 : Vector function에 관한 이야기이다!

1. f(alpha*x) = alpha * f(x)

2. f(x+y)=f(x) + f(y)

A vector function L is a linear transformation if

- You can scale first and then transform or transform first and then scale:

$$L(\alpha x) = \alpha L(x).$$

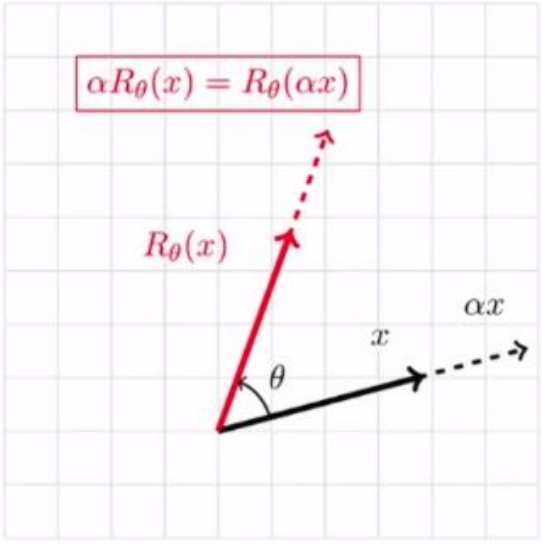
and

- You can transform first and then sum or sum first and then transform:

$$L(x + y) = L(x) + L(y).$$

Ex) 2D rotations or reflection

Let us examine a special property of this function:



You can rotate first and then scale or scale first and then rotate!

One can prove using high school geometry that

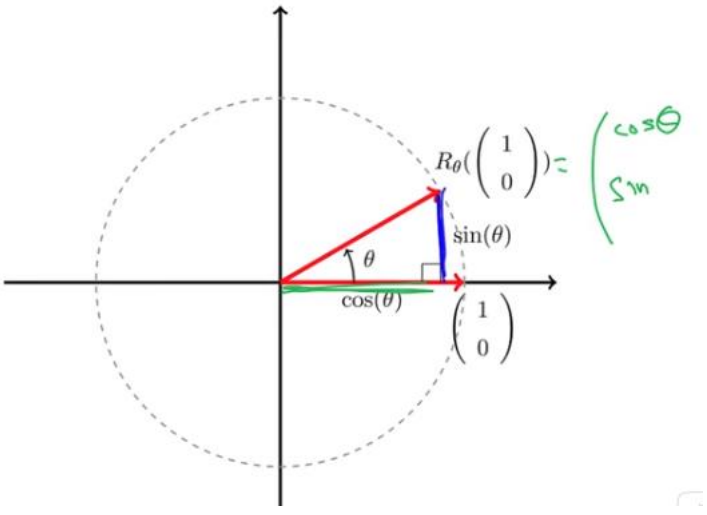
- You can stretch first and then rotate or rotate first and then stretch:

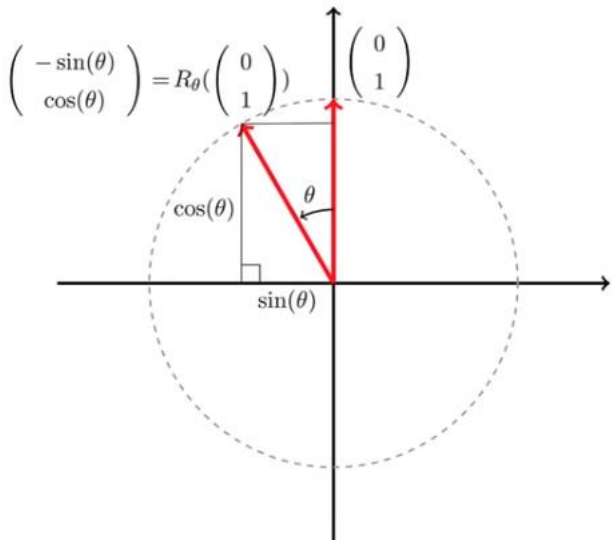
$$R_\theta(\alpha x) = \alpha R_\theta(x).$$

- You can sum first and then rotate or rotate first and then sum:

$$R_\theta(x + y) = R_\theta(x) + R_\theta(y).$$

Cf) rotation함수 구하기





→ unit basis vector를 넣어본다. 그러면 그 벡터를 합해서 A를 만들 수 있다.

$$
\begin{aligned}
R_\theta(x) = R_\theta(\begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix}) &= \underbrace{\left(\begin{array}{c|c} \alpha_{00} & \alpha_{01} \\ \alpha_{10} & \alpha_{11} \end{array}\right)}_{A} \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} \\
&= \left(R_\theta(\begin{pmatrix} 1 \\ 0 \end{pmatrix}) \middle| R_\theta(\begin{pmatrix} 0 \\ 1 \end{pmatrix}) \right) \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} \\
&= \left(\begin{array}{c|c} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{array}\right) \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} = \begin{pmatrix} \cos(\theta)\chi_0 - \sin(\theta)\chi_1 \\ \sin(\theta)\chi_0 + \cos(\theta)\chi_1 \end{pmatrix} \\
&= \chi_0 \begin{pmatrix} \cos(\theta) \\ \sin(\theta) \end{pmatrix} + \chi_1 \begin{pmatrix} -\sin(\theta) \\ \cos(\theta) \end{pmatrix}
\end{aligned}
$$

→ 이렇게 vector function을 $x_j a_j$의 합(linear combination)으로 표현할 수 있다.

Ex)

Example

The transformation $f(\begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix}) = \begin{pmatrix} \chi_0 + \chi_1 \\ \chi_0 \end{pmatrix}$ *is* a linear transformation.

The way we prove this is to pick arbitrary

$$\alpha \in \mathbb{R}, \quad x = \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix}, \quad \text{and} \quad y = \begin{pmatrix} \psi_0 \\ \psi_1 \end{pmatrix}.$$

with which we then show that

- $f(\alpha x) = \alpha f(x)$; and
- $f(x + y) = f(x) + f(y)$.

→ 증명하려면 손으로 증명해야 한다. 뒤에 두번째 방법도 있다. Linear transformation이라면 Ax일 것이다!

- Show $f(\alpha x) = \alpha f(x)$:

$$f(\alpha x) = f(\alpha \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix}) = f(\begin{pmatrix} \alpha\chi_0 \\ \alpha\chi_1 \end{pmatrix})$$

$$= \begin{pmatrix} \alpha\chi_0 + \alpha\chi_1 \\ \alpha\chi_0 \end{pmatrix} = \begin{pmatrix} \alpha(\chi_0 + \chi_1) \\ \alpha\chi_0 \end{pmatrix}$$

and

$$\alpha f(x) = \alpha f(\begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix}) = \alpha \begin{pmatrix} \chi_0 + \chi_1 \\ \chi_0 \end{pmatrix}$$

$$= \begin{pmatrix} \alpha(\chi_0 + \chi_1) \\ \alpha\chi_0 \end{pmatrix}$$

so that both $f(\alpha x)$ and $\alpha f(x)$ evaluate to the same expression.

(2) 또 다른 판정법

- vector function is a linear transformation ⇔ (if or only if) it could be represented by a matrix (Ax : matrix-vector multiplication)

Theorem

Let $L : \mathbb{R}^n \to \mathbb{R}^m$ be defined by $L(x) = Ax$ where $A \in \mathbb{R}^{m \times n}$.
Then L is a linear transformation.

Consequence:

A function $f : \mathbb{R}^n \to \mathbb{R}^m$ is a linear transformation if and only if it can be written as a matrix-vector multiplication.

예시)

Example 1

Alternate proof that $f(\begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix}) = \begin{pmatrix} \chi_0 + \chi_1 \\ \chi_0 \end{pmatrix}$ is a linear transformation.

- Compute a possible matrix that represents f:

  $f(\begin{pmatrix} 1 \\ 0 \end{pmatrix}) = \begin{pmatrix} 1+0 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$ and $f(\begin{pmatrix} 0 \\ 1 \end{pmatrix}) = \begin{pmatrix} 0+1 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$

- *if* f is a linear transformation, then $f(x) = Ax$ where $A = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}$.
- Now,

  $Ax = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} = \begin{pmatrix} \chi_0 + \chi_1 \\ \chi_0 \end{pmatrix}$

$f(x) = Ax$

- 완전 중요!!!! Linear transformation 혹은 vector function의 정체를 알려면 aj부터 알아야 한다. a_j란 unit basis vector를 vector function에 집어넣어서 변형된 결과를 의미한다.

The linear transformation L is completely described by the set of vectors $\{a_0, \ldots, a_{n-1}\}$, where $a_j = L(e_j)$

→ you don't need to know what the linear transformation is. L is completely described by how it transforms the unit basis vectors.

From linear transformation to matrix notation

Let $a_j = L(e_j)$. Arrange these vectors as the columns of a two dimensional array, A:

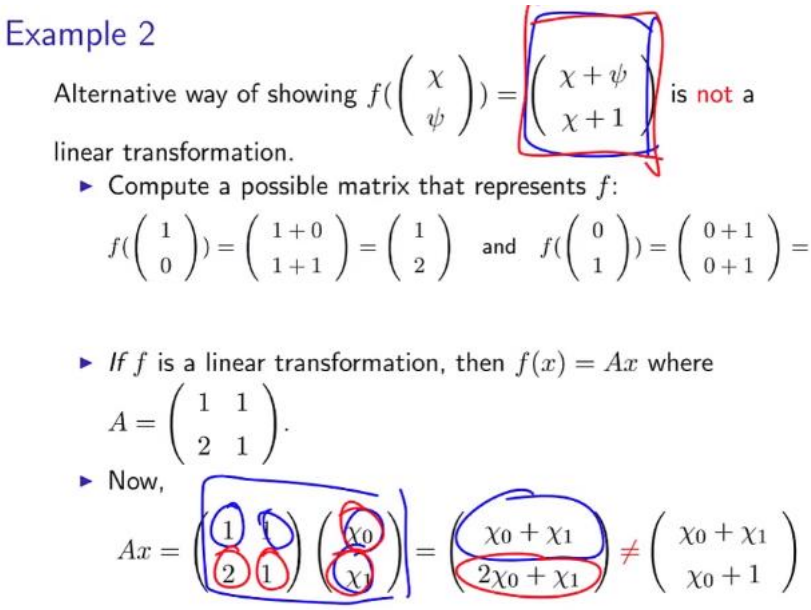
$$A = \left(\begin{array}{c|c|c|c} \alpha_{0,0} & \alpha_{0,1} & \cdots & \alpha_{0,n-1} \\ \alpha_{1,0} & \alpha_{1,1} & \cdots & \alpha_{1,n-1} \\ \vdots & \vdots & \ddots & \vdots \\ \alpha_{m-1,0} & \alpha_{m-1,1} & \cdots & \alpha_{m-1,n-1} \end{array} \right)$$

a_0 a_1 a_{n-1}

so that $\alpha_{i,j}$ equals the ith component of vector a_j.

- 풀이 : 따라서 vector function을 알려면 unit basis vector를 x로 넣어본다. 그러면 aj들을 조합해서

- 예시2)



Let f be a vector function such that $f(\begin{pmatrix} x_0 \\ x_1 \end{pmatrix}) = \begin{pmatrix} x_0^2 \\ x_1 \end{pmatrix}$ then

- (a) f is a linear transformation.
- (b) f is not a linear transformation. ✔
- (c) Not enough information is given to determine whether or not f is a linear transformation.

Explanation

Answer: (b): To compute a possible matrix that represents f consider:

$$f(\begin{pmatrix} 1 \\ 0 \end{pmatrix}) = \begin{pmatrix} 1^2 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \quad \text{and} \quad f(\begin{pmatrix} 0 \\ 1 \end{pmatrix}) = \begin{pmatrix} 0^2 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}.$$

Thus, *if* f is a linear transformation, then $f(x) = Ax$ where $A = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$. Now,

$$Ax = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} x_0 \\ x_1 \end{pmatrix} = \begin{pmatrix} x_0 \\ x_1 \end{pmatrix} \neq \begin{pmatrix} x_0^2 \\ x_1 \end{pmatrix} = f(\begin{pmatrix} x_0 \\ x_1 \end{pmatrix}) = f(x).$$

→ 이것도 unit basis vector를 넣어서 어떻게 바뀌는지 보고 그 벡터들을 옆으로 나란히 붙여서 행렬 A로 만든다!!

2.2. 중요성

F는 vector를 input으로 vector가 output으로 나오는 함수다.

(1) f 를 알고 y를 알 때, 해 x를 구하고 싶다.

(2) f를 아는데, 이를 스칼라랑 x의 곱으로 표현하고 싶다!!! ➔ Eigen value problem

→ f가 linear transformation일 때, 이런 문제가 쉽게 풀린다. (linear가 아니면 local 구간을 정해서 선형으로

근사를 한다.)

➔ linear transformations are a subset of vector functions for which these problems are simpler to solve.

An archtypical problem in science and engineering:

$$f : \mathbb{R}^n \to \mathbb{R}^m.$$

Typical questions that we want to answer:

- Given vector $x \in \mathbb{R}^n$, evaluate $f(x)$, or
- Given vector $y \in \mathbb{R}^m$, find x such that $f(x) = y$; or
- Find scalar λ and vector x such that $f(x) = \lambda x$ (only if $m = n$).

2.3. 파생 특징과 증명

Let $v_0, v_1, \ldots, v_{k-1} \in \mathbb{R}^n$ and let $L : \mathbb{R}^n \to \mathbb{R}^m$ be a linear transformation.

Then

$$L(v_0 + v_1 + \ldots + v_{k-1}) = L(v_0) + L(v_1) + \ldots + L(v_{k-1}).$$

증명 : Proof by induction (귀납법)

1) k=1일 때 성립

2) k=K일 때 성립하면 k=K+1일 때 성립한다는 것을 증명

Inductive step (continued):

$$
\begin{aligned}
& L(v_0 + v_1 + \ldots + v_K) \\
= \quad & \text{< expose extra term – We know we can do this, since } K \geq 1 > \\
& L(v_0 + v_1 + \ldots + v_{K-1} + v_K) \\
= \quad & \text{< associativity of vector addition >} \\
& L((v_0 + v_1 + \ldots + v_{K-1}) + v_K) \\
= \quad & \text{< } L \text{ is a linear transformation) >} \\
& L(v_0 + v_1 + \ldots + v_{K-1}) + L(v_K) \\
= \quad & \text{< Inductive Hypothesis >} \\
& L(v_0) + L(v_1) + \ldots + L(v_{K-1}) + L(v_K)
\end{aligned}
$$

3) By the principle of mathematical induction, the result holds for all k.

If one can show that

- (Base case) a property holds for $k = k_b$; *and*
- (Inductive step) if it holds for $k = K$, where $K \geq k_b$, then it is also holds for $k = K + 1$,

then one can conclude that the property holds for all integers $k \geq k_b$.

Often $k_b = 0$ or $k_b = 1$.

- induction이 중요한 이유 :

> *Dijkstra: "Today a usual technique is to make a program and then to test it. But: program testing can be a very effective way to show the presence of bugs, but is hopelessly inadequate for showing their absence. The only effective way to raise the confidence level of a program significantly is to give a convincing proof of its correctness.* **But one should not first make the program and then prove its correctness, because then the requirement of providing the proof would only increase the poor programmers burden. On the contrary: the programmer should let correctness proof and program grow hand in hand."**

→ linear algebra는 loop-based program이 많다. 그런데 이를 검증할 때 쉽지 않다. 이를 하기 위해서 두 가지가 필요하다. 1) slicing and dicing 2) mathematical induction → proves the correctness of a loop

2.4. Matrix : representing linear transformation as matrices

- matrix를 사용하는 이유 : matrix is a convenient way of representing linear transformation

Let $\{v_o, v_1, \ldots, v_{k-1}\} \in \mathbb{R}^n$, $\{\alpha_o, \alpha_1, \ldots, \alpha_{k-1}\} \in \mathbb{R}$, and let $L : \mathbb{R}^n \to \mathbb{R}^m$ be a linear transformation. Then

$$L(\alpha_0 v_0 + \alpha_1 v_1 + \ldots + \alpha_{k-1} v_{k-1}) = \alpha_0 L(v_0) + \alpha_1 L(v_1) + \ldots + \alpha_{k-1} L(v_{k-1}).$$

- We already saw that if L is a linear transformation, then $L(\alpha x + \beta y) = \alpha L(x) + \beta L(y)$.

More concise notation:

$$L(\sum_{j=0}^{k-1} \alpha_j v_j) = \sum_{j=0}^{k-1} \alpha_j L(v_j)$$

→ vector들의 linear combination을 transform하거나, transform first then take the linear combination하는 것이 똑같다는 것이다.

참고로) 모든 vector x는 linear combination of unit basis vector로 나타낼 수 있다는 사실을 꼭 기억하라

Recall: any $x \in \mathbb{R}^n$ can be written as

$$x = \begin{pmatrix} \chi_0 \\ \chi_1 \\ \vdots \\ \chi_{n-1} \end{pmatrix} = \chi_0 \underbrace{\begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix}}_{e_0} + \chi_1 \underbrace{\begin{pmatrix} 0 \\ 1 \\ \vdots \\ 0 \end{pmatrix}}_{e_1} + \cdots + \chi_{n-1} \underbrace{\begin{pmatrix} 0 \\ 0 \\ \vdots \\ 1 \end{pmatrix}}_{e_{n-1}} = \sum_{j=0}^{n-1} \chi_j e_j.$$

→ vector의 component가 coefficient가 되는 것이다.

$$
\begin{aligned}
y &= L(x) && < \text{Reason} > \\
&= L\left(\sum_{j=0}^{n-1} \chi_j e_j\right) && < x = \sum_{j=0}^{n-1} \chi_j e_j > \\
&= \sum_{j=0}^{n-1} \chi_j L(e_j) && < L(\sum_{j=0}^{n-1} \alpha_j v_j) = \sum_{j=0}^{n-1} \alpha_j L(v_j) > \\
&= \sum_{j=0}^{n-1} \chi_j \underbrace{L(e_j)}_{a_j} && < \text{Let } a_j = L(e_j) > \\
&= \sum_{j=0}^{n-1} \chi_j a_j.
\end{aligned}
$$

→ L(x)가 x vector의 linear combination이다. 따라서 이 것의 효과는 if you know what those vectors(a_j) are, then you know how to evaluate L(x). (L(x)를 모르는 상황이더라도!!)

➔ The action of linear transformation is completely described by how it transforms the unit basis vectors

➔ Evaluating the linear transformation is equivalent to performing a matrix-vector multiplication (Ax) as you already knew it. L(x) = Ax...와..... 대박....

alternatively

Less concisely:

$$
\begin{aligned}
L(x) &= L(\chi_0 e_0 + \chi_1 e_1 + \cdots + \chi_{n-1} e_{n-1}) \\
&= \chi_0 a_0 + \chi_1 a_1 + \cdots + \chi_{n-1} a_{n-1},
\end{aligned}
$$

where $a_j = L(e_j)$.

- 완전 중요!!!!

The linear transformation L is completely described by the set of vectors $\{a_0, \ldots, a_{n-1}\}$, where $a_j = L(e_j)$

→ you don't need to know what the linear transformation is. L is completely described by how it transforms the unit basis vectors.

From linear transformation to matrix notation

Let $a_j = L(e_j)$. Arrange these vectors as the columns of a two dimensional array, A:

$$A = \begin{pmatrix} \alpha_{0,0} & \alpha_{0,1} & \cdots & \alpha_{0,n-1} \\ \alpha_{1,0} & \alpha_{1,1} & \cdots & \alpha_{1,n-1} \\ \vdots & \vdots & \ddots & \vdots \\ \alpha_{m-1,0} & \alpha_{m-1,1} & \cdots & \alpha_{m-1,n-1} \end{pmatrix}$$

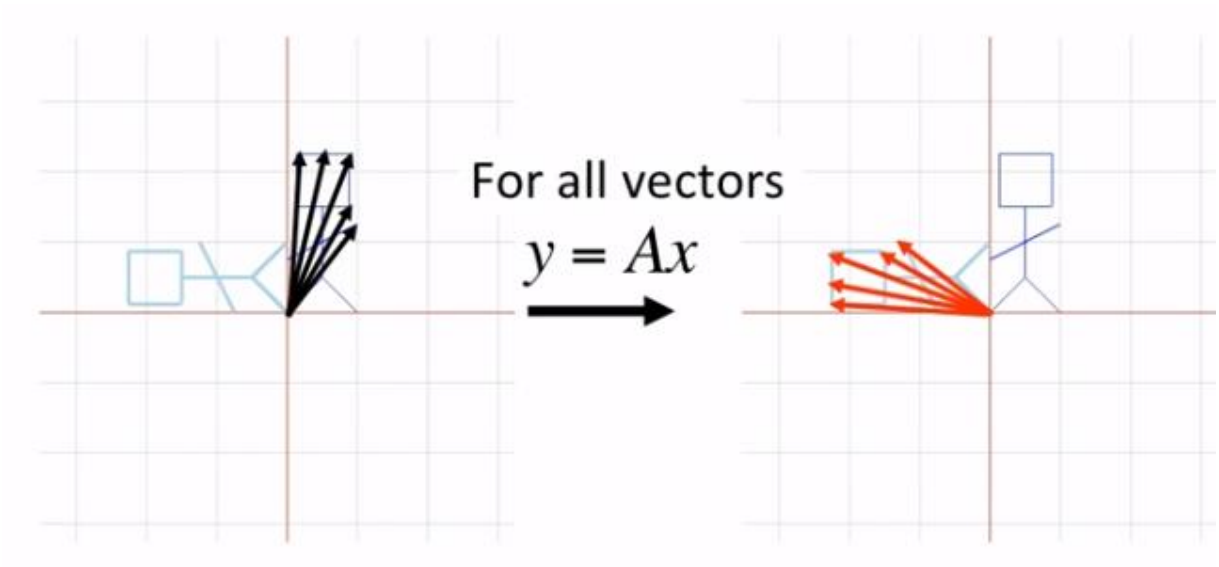
a_0 a_1 a_{n-1}

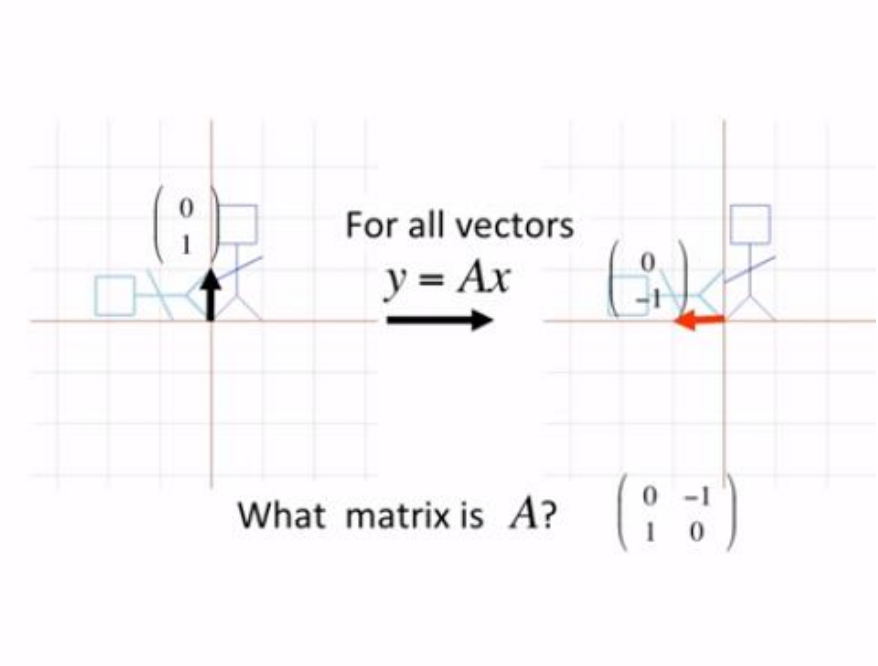
so that $\alpha_{i,j}$ equals the ith component of vector a_j.

➔ 와 이 강의에서 제일 중요한 말인 것 같다.

다시 보기 : https://courses.edx.org/courses/course-v1:UTAustinX+UT.5.05x+2T2017/courseware/b39776b93e54417bbd359e40b15dbfed/a97b0caa357241c280ec5b39a4c0110c/?child=first

- 예시 : Timmy Two space





→ unit basis vector를 어떻게 변형시키는지 보면 쉽게 찾을 수 있다.

- Vector a_j들로서 linear transformation을 표현/나타낼 수 있다. 다시 a_j들은 하나의 행렬로 예쁘고 정갈하게 나타낼 수 있다. 따라서, matrix is a convenient way to store information about the linear transformation.

So, now A represents the linear transformation L.

We will write $L(x) = Ax$.

How is Ax then computed?

$$
\begin{aligned}
Ax = L(x) &= \chi_0 a_0 + \chi_1 a_1 + \cdots + \chi_{n-1} a_{n-1} \\
&= \chi_0 \begin{pmatrix} \alpha_{0,0} \\ \alpha_{1,0} \\ \vdots \\ \alpha_{m-1,0} \end{pmatrix} + \chi_1 \begin{pmatrix} \alpha_{0,1} \\ \alpha_{1,1} \\ \vdots \\ \alpha_{m-1,1} \end{pmatrix} + \cdots + \chi_{n-1} \begin{pmatrix} \alpha_{0,n-1} \\ \alpha_{1,n-1} \\ \vdots \\ \alpha_{m-1,n-1} \end{pmatrix} \\
&= \begin{pmatrix} \chi_0\alpha_{0,0} + & \chi_1\alpha_{0,1} + & \cdots + & \chi_{n-1}\alpha_{0,n-1} \\ \chi_0\alpha_{1,0} + & \chi_1\alpha_{1,1} + & \cdots + & \chi_{n-1}\alpha_{1,n-1} \\ \vdots & \vdots & \vdots & \vdots \\ \chi_0\alpha_{m-1,0} + & \chi_1\alpha_{m-1,1} + & \cdots + & \chi_{n-1}\alpha_{m-1,n-1} \end{pmatrix} \\
&= \begin{pmatrix} \alpha_{0,0}\chi_0 + & \alpha_{0,1}\chi_1 + & \cdots + & \alpha_{0,n-1}\chi_{n-1} \\ \alpha_{1,0}\chi_0 + & \alpha_{1,1}\chi_1 + & \cdots + & \alpha_{1,n-1}\chi_{n-1} \\ \vdots & \vdots & \vdots & \vdots \\ \alpha_{m-1,0}\chi_0 + & \alpha_{m-1,1}\chi_1 + & \cdots + & \alpha_{m-1,n-1}\chi_{n-1} \end{pmatrix}.
\end{aligned}
$$

Let $A \in \mathbb{R}^{mxn}; x, y \in \mathbb{R}^n;$ and $\alpha \in \mathbb{R}$. Then

$A(\alpha x) = \alpha Ax.$

$A(x + y) = Ax + Ay$

In other words, matrix-vector multiplication is a linear transformation.

3. Matrix-Vector operations

Special Matrices

3.2.1 The Zero Matrix

3.2.2 The Identity Matrix

3.2.3 Diagonal Matrices

3.2.4 Triangular Matrices

3.2.5 Transpose Matrix

3.2.6 Symmetric Matrices

Operations with Matrices

3.3.1 Scaling a Matrix

3.3.2 Adding Matrices

Matrix-Vector Multiplication Algorithms

3.4.1 Via Dot Products

3.4.2 Via AXPY Operations

3.4.3 Compare and Contrast

3.4.4 Cost of Matrix-Vector Multiplication

3. Matrix-Vector operations

3.1. Matrix의 종류

3.1.1. Zero matrix

- 이 vector function은 무엇일까? Linear transformation일까?

Let $L_0 : \mathbb{R}^3 \to \mathbb{R}^2$ be the function defined for every $x \in \mathbb{R}^3$ by $L_0(x) = 0$, where 0 denotes the zero vector "of appropriate size":

$$L_0\left(\begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix}\right) = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

Is this a linear transformation? If it is, what matrix represents it?

- 해설 : unit basis vector를 넣어본다. 순차적으로 a_j를 모은 것이 바로 A이다. 만약 A가 이 vector function을 잘 대변할 수 있다면 이 벡터함수는 linear transformation이라고 할 수 있다.

$$L_0\left(\begin{pmatrix}1\\0\\0\end{pmatrix}\right)=\begin{pmatrix}0\\0\end{pmatrix}, L_0\left(\begin{pmatrix}0\\1\\0\end{pmatrix}\right)=\begin{pmatrix}0\\0\end{pmatrix}, L_0\left(\begin{pmatrix}0\\0\\1\end{pmatrix}\right)=\begin{pmatrix}0\\0\end{pmatrix}$$

- *If* this is a linear transformation, then the matrix

$$\begin{pmatrix}0&0&0\\0&0&0\end{pmatrix}$$

represents it.

- But

$$L_0\left(\begin{pmatrix}\chi_0\\\chi_1\\\chi_2\end{pmatrix}\right)=\begin{pmatrix}0\\0\end{pmatrix}=\begin{pmatrix}0&0&0\\0&0&0\end{pmatrix}\begin{pmatrix}\chi_0\\\chi_1\\\chi_2\end{pmatrix}.$$

- So...

The $m \times n$ matrix

$$\begin{pmatrix}0&0&\cdots&0\\0&0&\cdots&0\\\vdots&\vdots&&\vdots\\0&0&\cdots&0\end{pmatrix}$$

will be denoted by 0 or $0_{m\times n}$. It is called the zero matrix.

3.1.2. Identity matrix : vector가 들어가면 똑 같은 vector가 나온다!!

- 문제 :

Let $L_I : \mathbb{R}^3 \to \mathbb{R}^3$ be the function defined for every $x \in \mathbb{R}^3$ by $L_I(x) = x$: appropriate size":

$$L_I\left(\begin{pmatrix}\chi_0\\\chi_1\\\chi_2\end{pmatrix}\right)=\begin{pmatrix}\chi_0\\\chi_1\\\chi_2\end{pmatrix}$$

Is this a linear transformation? If it is, what matrix represents it?

- 어떻게 아는가? : unit basis vector를 집어넣어 본다!!

$$L_I\left(\begin{pmatrix}1\\0\\0\end{pmatrix}\right)=\begin{pmatrix}1\\0\\0\end{pmatrix}, L_I\left(\begin{pmatrix}0\\1\\0\end{pmatrix}\right)=\begin{pmatrix}0\\1\\0\end{pmatrix}, L_I\left(\begin{pmatrix}0\\0\\1\end{pmatrix}\right)=\begin{pmatrix}0\\0\\1\end{pmatrix}$$

- *If* this is a linear transformation, then the matrix

$$\left(\begin{array}{c|c|c}1&0&0\\0&1&0\\0&0&1\end{array}\right)$$

represents it.

- But

$$L_I\left(\begin{pmatrix}\chi_0\\\chi_1\\\chi_2\end{pmatrix}\right)=\begin{pmatrix}\chi_0\\\chi_1\\\chi_2\end{pmatrix} \stackrel{?}{=} \begin{pmatrix}1&0&0\\0&1&0\\0&0&1\end{pmatrix}\begin{pmatrix}\chi_0\\\chi_1\\\chi_2\end{pmatrix}.$$

→ A를 실제로 해보니, 실제로 그렇게 되는구나!! 와 identity matrix를 이렇게 생각해본 적은 여태까지 없었다!!

Summary

The $n \times n$ matrix

$$\begin{pmatrix}1&0&\cdots&0\\0&1&\cdots&0\\\vdots&\ddots&&\vdots\\0&0&\cdots&1\end{pmatrix}$$

will be denoted by I or I_n. It is called the identity matrix.

$$Ix = x$$

for any vector x of size n.

3.1.3. Diagonal matrix

- 기능에서 시작

Let $L_D : \mathbb{R}^3 \to \mathbb{R}^3$ be the function defined by

$$L_D\left(\begin{pmatrix}\chi_0\\\chi_1\\\chi_2\end{pmatrix}\right)=\begin{pmatrix}\delta_0\chi_0\\\delta_1\chi_1\\\delta_2\chi_2\end{pmatrix}$$

Is this a linear transformation? If it is, what matrix represents it?

- 해결 과정 :

$$L_D\left(\begin{pmatrix}1\\0\\0\end{pmatrix}\right)=\begin{pmatrix}\delta_0\\0\\0\end{pmatrix}, L_D\left(\begin{pmatrix}0\\1\\0\end{pmatrix}\right)=\begin{pmatrix}0\\\delta_1\\0\end{pmatrix}, L_D\left(\begin{pmatrix}0\\0\\1\end{pmatrix}\right)=\begin{pmatrix}0\\0\\\delta_2\end{pmatrix}.$$

- *If* this is a linear transformation, then the matrix

$$\begin{pmatrix}\delta_0 & 0 & 0\\0 & \delta_1 & 0\\0 & 0 & \delta_2\end{pmatrix}$$

- linear transformation이다!!

represents it.

- But

$$L_D\left(\begin{pmatrix}\chi_0\\\chi_1\\\chi_2\end{pmatrix}\right)=\begin{pmatrix}\delta_0\chi_0\\\delta_1\chi_1\\\delta_2\chi_2\end{pmatrix} \overset{?}{=} \begin{pmatrix}\delta_0 & 0 & 0\\0 & \delta_1 & 0\\0 & 0 & \delta_2\end{pmatrix}\begin{pmatrix}\chi_0\\\chi_1\\\chi_2\end{pmatrix}.$$

- 결론 :

An $n \times n$ matrix of the form

$$\begin{pmatrix}\delta_0 & 0 & \cdots & 0\\0 & \delta_1 & \cdots & 0\\\vdots & \vdots & \ddots & \vdots\\0 & 0 & \cdots & \delta_{n-1}\end{pmatrix}$$

is called a diagonal matrix.

3.1.4. Triangular matrices

- 문제 :

Let $L_U : \mathbb{R}^3 \to \mathbb{R}^3$ be the function defined by

$$L_U\left(\begin{pmatrix}\chi_0\\\chi_1\\\chi_2\end{pmatrix}\right)=\begin{pmatrix}2\chi_0-\chi_1+\chi_2\\3\chi_1-\chi_2\\-2\chi_2\end{pmatrix}$$

Is this a linear transformation? If it is, what matrix represents it?

- 해결 : unit basis vector를 넣고, 나온 결과를 aj라고 하면, 그것을 모은 것이 A matrix이고, linear transformation이라면 Ax = L(x) 인지만 확인하면 된다.

- upper triangular matrix

- strictly upper triangular matrix : 대각선을 포함한 아랫부분이 다 0

- unit upper triangular matrix : 대각선이 1이고, 그 밑부분은 0.

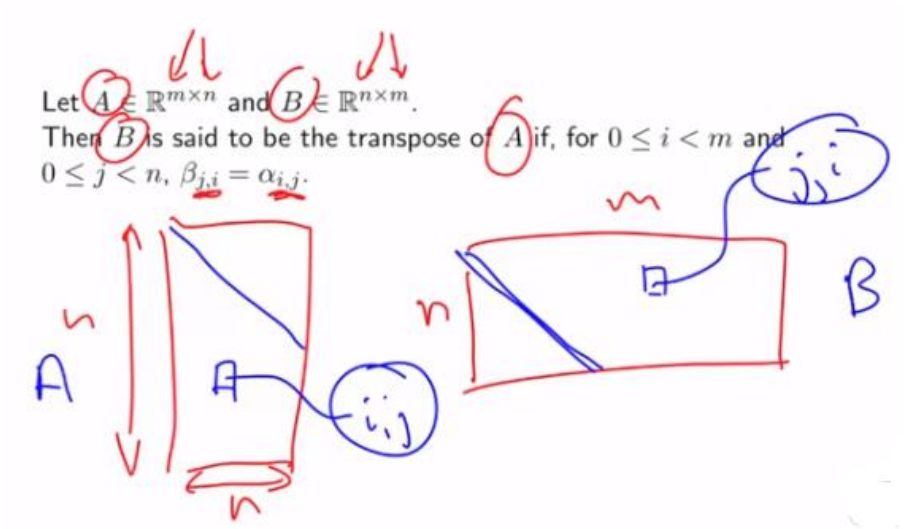
- lower triangular matrix :

- strictly lower triangular matrix : 대각선을 포함한 윗부분이 다 0

- unit lower triangular matrix : 대각선이 1이고, 그 윗부분은 0.

3.1.5. Transpose matrix

- 정의 : mirroring diagonally!!



$$A = \begin{pmatrix} \alpha_{0,0} & \alpha_{0,1} & \alpha_{0,2} & \alpha_{0,3} \\ \alpha_{1,0} & \alpha_{1,1} & \alpha_{1,2} & \alpha_{1,3} \\ \alpha_{2,0} & \alpha_{2,1} & \alpha_{2,2} & \alpha_{2,3} \end{pmatrix}$$

$$B = A^T = \begin{pmatrix} \alpha_{0,0} & \alpha_{1,0} & \alpha_{2,0} \\ \alpha_{0,1} & \alpha_{1,1} & \alpha_{2,1} \\ \alpha_{0,2} & \alpha_{1,2} & \alpha_{2,2} \\ \alpha_{0,3} & \alpha_{1,3} & \alpha_{2,3} \end{pmatrix}$$

3.1.6. Symmetric matrix

- 조건 : R^{nxn}이어야 한다. $A^T=A$

- $\alpha_{ij}=\alpha_{ji}$

3.2. Matrix-vector multiplication 연산

3.2.1. 곱과 합

(1) Scaling a matrix

- $L_B(x) = \beta L_A(x)$

→ $L_B(x) = \beta Ax$

Now, assume that matrix $A \in \mathbb{R}^{2\times 3}$ represents linear transformation L_A:

$$L_A(x) = Ax \quad \text{and} \quad A = \begin{pmatrix} \alpha_{0,0} & \alpha_{0,1} & \alpha_{0,2} \\ \alpha_{1,0} & \alpha_{1,1} & \alpha_{1,2} \end{pmatrix}.$$

Then

$$L_B(\begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}) = \beta L(\begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}) = \beta \begin{pmatrix} \alpha_{0,0} \\ \alpha_{1,0} \end{pmatrix} = \begin{pmatrix} \beta\alpha_{0,0} \\ \beta\alpha_{1,0} \end{pmatrix}$$

$$L_B(\begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}) = \beta L(\begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}) = \beta \begin{pmatrix} \alpha_{0,1} \\ \alpha_{1,1} \end{pmatrix} = \begin{pmatrix} \beta\alpha_{0,1} \\ \beta\alpha_{1,1} \end{pmatrix}$$

$$L_B(\begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}) = \beta L(\begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}) = \beta \begin{pmatrix} \alpha_{0,2} \\ \alpha_{1,2} \end{pmatrix} = \begin{pmatrix} \beta\alpha_{0,2} \\ \beta\alpha_{1,2} \end{pmatrix}$$

(2) Adding matrices

Given $L_A : \mathbb{R}^3 \to \mathbb{R}^2$, and $L_B : \mathbb{R}^3 \to \mathbb{R}^2$ let $L_C : \mathbb{R}^3 \to \mathbb{R}^2$ be the function defined for every $x \in \mathbb{R}^3$ by $L_C(x) = L_A(x) + L_B(x)$.

(Hence $L_C(x) = Ax + Bx$, where A is the matrix that represents L_A and B is the matrix that represents L_B.)

$$L_C(\begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix}) = L_A(\begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix}) + L_B(\begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix})$$

Is this a linear transformation? If it is, what matrix represents it?

→ linear transformation의 관점에서 보자. Unit basis vector를 어떻게 변형시키는 지를 보고(a_j) 그것을 모으면 A가 된다. 이는 $L_A(x)$를 represent한다.

Matrix $A \in \mathbb{R}^{2\times 3}$ represents linear transformation L_A:

$$L_A(x) = Ax \quad \text{and} \quad A = \left(\begin{array}{c|c|c} \alpha_{0,0} & \alpha_{0,1} & \alpha_{0,2} \\ \alpha_{1,0} & \alpha_{1,1} & \alpha_{1,2} \end{array}\right).$$

Matrix $B \in \mathbb{R}^{2\times 3}$ represents linear transformation L_B:

$$L_B(x) = Bx \quad \text{and} \quad B = \left(\begin{array}{c|c|c} \beta_{0,0} & \beta_{0,1} & \beta_{0,2} \\ \beta_{1,0} & \beta_{1,1} & \beta_{1,2} \end{array}\right).$$

Then

$$L_C\left(\begin{pmatrix}1\\0\\0\end{pmatrix}\right) = L_A\left(\begin{pmatrix}1\\0\\0\end{pmatrix}\right) + L_B\left(\begin{pmatrix}1\\0\\0\end{pmatrix}\right) = \begin{pmatrix}\alpha_{0,0}\\\alpha_{1,0}\end{pmatrix} + \begin{pmatrix}\beta_{0,0}\\\beta_{1,0}\end{pmatrix} = \begin{pmatrix}\alpha_{0,0}+\beta_{0,0}\\\alpha_{1,0}+\beta_{1,0}\end{pmatrix}$$

$$L_C\left(\begin{pmatrix}0\\1\\0\end{pmatrix}\right) = L_A\left(\begin{pmatrix}0\\1\\0\end{pmatrix}\right) + L_B\left(\begin{pmatrix}0\\1\\0\end{pmatrix}\right) = \begin{pmatrix}\alpha_{0,1}\\\alpha_{1,1}\end{pmatrix} + \begin{pmatrix}\beta_{0,1}\\\beta_{1,1}\end{pmatrix} = \begin{pmatrix}\alpha_{0,1}+\beta_{0,1}\\\alpha_{1,1}+\beta_{1,1}\end{pmatrix}$$

$$L_C\left(\begin{pmatrix}0\\0\\1\end{pmatrix}\right) = L_A\left(\begin{pmatrix}0\\0\\1\end{pmatrix}\right) + L_B\left(\begin{pmatrix}0\\0\\1\end{pmatrix}\right) = \begin{pmatrix}\alpha_{0,2}\\\alpha_{1,2}\end{pmatrix} + \begin{pmatrix}\beta_{0,2}\\\beta_{1,2}\end{pmatrix} = \begin{pmatrix}\alpha_{0,2}+\beta_{0,2}\\\alpha_{1,2}+\beta_{1,2}\end{pmatrix}$$

3.2. Matrix-vector multiplication의 알고리즘

(1) Row단위로 하면 Dot product 방식 → row 단위로 해도 되고, diagonal하게 잘라서 row를 삼등분해서 해도 되고(그렇게 하면 triangular, symmetric할 때 유용하다!)

(2) Column단위로 하면 linear combination 방식 → column 단위로 해도 되고, diagonal하게 잘라서 column를 삼등분해서 해도 되고(그렇게 하면 triangular, symmetric할 때 유용하다!)

3.2.1. Matrix를 행으로 만들어버려! ➔ Dot product로!

➔ Matrix-vector multiplication을 좀 다르게 생각하는 방법 (Dot product로 생각!)

- Matrix-vector multiplication can be thought of as inner products of rows of the matrix with the vector being multiplied. ➔ dot product로 표현할 수 있다는 것이 중요하다!

$$\begin{pmatrix}\psi_0\\\psi_1\\\psi_2\\\psi_3\end{pmatrix} := \begin{pmatrix}\alpha_{0,0} & \alpha_{0,1} & \alpha_{0,2}\\\alpha_{1,0} & \alpha_{1,1} & \alpha_{1,2}\\\alpha_{2,0} & \alpha_{2,1} & \alpha_{2,2}\\\alpha_{3,0} & \alpha_{3,1} & \alpha_{3,2}\end{pmatrix}\begin{pmatrix}\chi_0\\\chi_1\\\chi_2\end{pmatrix} = \begin{pmatrix}\widetilde{a}_0^T\\\widetilde{a}_1^T\\\widetilde{a}_2^T\\\widetilde{a}_3^T\end{pmatrix} x = \begin{pmatrix}\widetilde{a}_0^T x\\\widetilde{a}_1^T x\\\widetilde{a}_2^T x\\\widetilde{a}_3^T x\end{pmatrix}$$

$$\begin{pmatrix} \psi_0 \\ \psi_1 \\ \psi_2 \\ \psi_3 \end{pmatrix} +:= \begin{pmatrix} \alpha_{0,0} & \alpha_{0,1} & \alpha_{0,2} \\ \alpha_{1,0} & \alpha_{1,1} & \alpha_{1,2} \\ \alpha_{2,0} & \alpha_{2,1} & \alpha_{2,2} \\ \alpha_{3,0} & \alpha_{3,1} & \alpha_{3,2} \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix} = \begin{pmatrix} \widetilde{a}_0^T \\ \widetilde{a}_1^T \\ \widetilde{a}_2^T \\ \widetilde{a}_3^T \end{pmatrix} x = \begin{pmatrix} \widetilde{a}_0^T x \\ \widetilde{a}_1^T x \\ \widetilde{a}_2^T x \\ \widetilde{a}_3^T x \end{pmatrix}$$

for $i = 0, \ldots, m-1$
  for $j = 0, \ldots, n-1$
    $\psi_i := \psi_i + \alpha_{i,j}\chi_j$
  endfor
  $\psi_i := \psi_i + \widetilde{a}_i^T x$
endfor

→ 요런 식으로!

y^{cur}	A	x	y^{next}
$\begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$	$\begin{pmatrix} 1 & -1 & 3 & 2 & -2 \\ 2 & -2 & 1 & 0 & -1 \\ 0 & -4 & 3 & 2 & 1 \\ 3 & 1 & -2 & 1 & 0 \\ -1 & 2 & 1 & -1 & -2 \end{pmatrix}$	$\begin{pmatrix} -1 \\ 0 \\ 2 \\ -1 \\ 1 \end{pmatrix}$	$\begin{pmatrix} 1 \\ -1 \\ 0 \\ 0 \\ 0 \end{pmatrix}$

$$\begin{pmatrix} 2 & -2 & 1 & 0 & -1 \end{pmatrix} \begin{pmatrix} -1 \\ 0 \\ 2 \\ -1 \\ 1 \end{pmatrix} = -1$$

- Alternative way : 이런 식으로... 대각선으로 내려가면서 대각선 양 옆의 row vector를 곱해서 dot product를 더하면 된다.

y^{cur}	A	x	y^{next}
$\begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$	$\begin{pmatrix} 1 & -1 & 3 & 2 & -2 \\ 2 & -2 & 1 & 0 & -1 \\ 0 & -4 & 3 & 2 & 1 \\ 3 & 1 & -2 & 1 & 0 \\ -1 & 2 & 1 & -1 & -2 \end{pmatrix}$	$\begin{pmatrix} -1 \\ 0 \\ 2 \\ -1 \\ 1 \end{pmatrix}$	$\begin{pmatrix} 1 \\ -1 \\ 0 \\ 0 \\ 0 \end{pmatrix}$

$$\begin{pmatrix} 2 \end{pmatrix} \begin{pmatrix} -1 \end{pmatrix} + (-2) \times (0) + \begin{pmatrix} 1 & 0 & -1 \end{pmatrix} \begin{pmatrix} 2 \\ -1 \\ 1 \end{pmatrix} = -1$$

→ triangular matrix일 때 참 유용하다.

Why is this alternative way important?
Consider the case where A is lower triangular.

y^{cur}	A	x	y^{next}
$\begin{pmatrix} -1 \\ -2 \\ \hline 0 \\ \hline 0 \\ 0 \end{pmatrix}$	$\left(\begin{array}{cc\|c\|cc} 1 & 0 & 0 & 0 & 0 \\ 2 & -2 & 0 & 0 & 0 \\ \hline 0 & -4 & 3 & 0 & 0 \\ \hline 3 & 1 & -2 & 1 & 0 \\ -1 & 2 & 1 & -1 & -2 \end{array}\right)$	$\begin{pmatrix} -1 \\ 0 \\ \hline 2 \\ \hline -1 \\ 1 \end{pmatrix}$	$\begin{pmatrix} -1 \\ -2 \\ \hline 6 \\ \hline 0 \\ 0 \end{pmatrix}$

$$\begin{pmatrix} 0 & -4 \end{pmatrix}\begin{pmatrix} -1 \\ 0 \end{pmatrix} + (3) \times (2) + \underbrace{\begin{pmatrix} 0 & 0 \end{pmatrix}\begin{pmatrix} -1 \\ 1 \end{pmatrix}}_{\text{Always 0!!!}} = 6$$

→ triangular!!!!

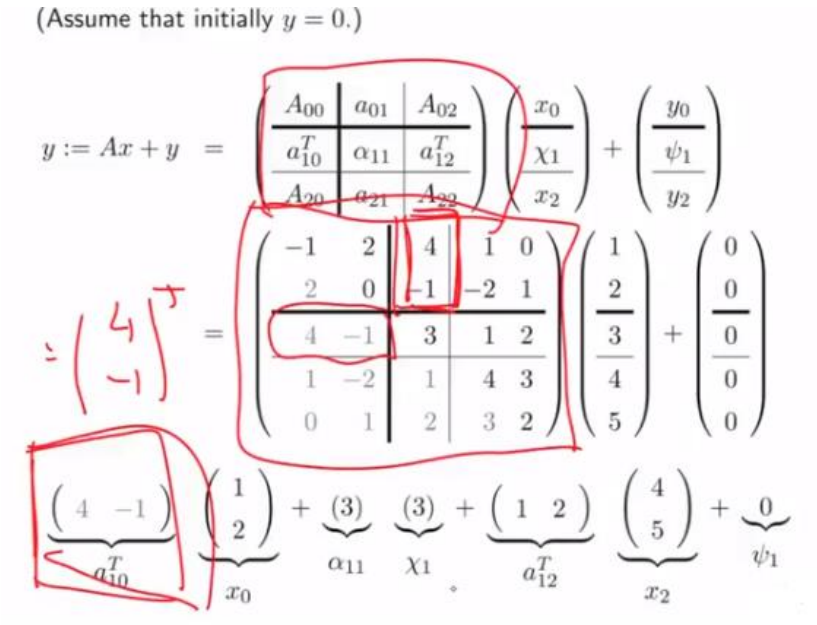
(Assume that initially $y = 0$.)

$$y := Ux + y = \left(\begin{array}{cc\|c\|cc} U_{00} & & u_{01} & & U_{02} \\ & & & & \\ \hline u_{10}^T & & \upsilon_{11} & & u_{12}^T \\ \hline U_{20} & & u_{21} & & U_{22} \end{array}\right)\begin{pmatrix} x_0 \\ \hline \chi_1 \\ \hline x_2 \end{pmatrix} + \begin{pmatrix} y_0 \\ \hline \psi_1 \\ \hline y_2 \end{pmatrix}$$

$$= \left(\begin{array}{cc\|c\|cc} -1 & 2 & 4 & 1 & 0 \\ 0 & 0 & -1 & -2 & 1 \\ \hline 0 & 0 & 3 & 1 & 2 \\ \hline 0 & 0 & 0 & 4 & 3 \\ 0 & 0 & 0 & 0 & 2 \end{array}\right)\begin{pmatrix} 1 \\ 2 \\ \hline 3 \\ \hline 4 \\ 5 \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ \hline 0 \\ \hline 0 \\ 0 \end{pmatrix}$$

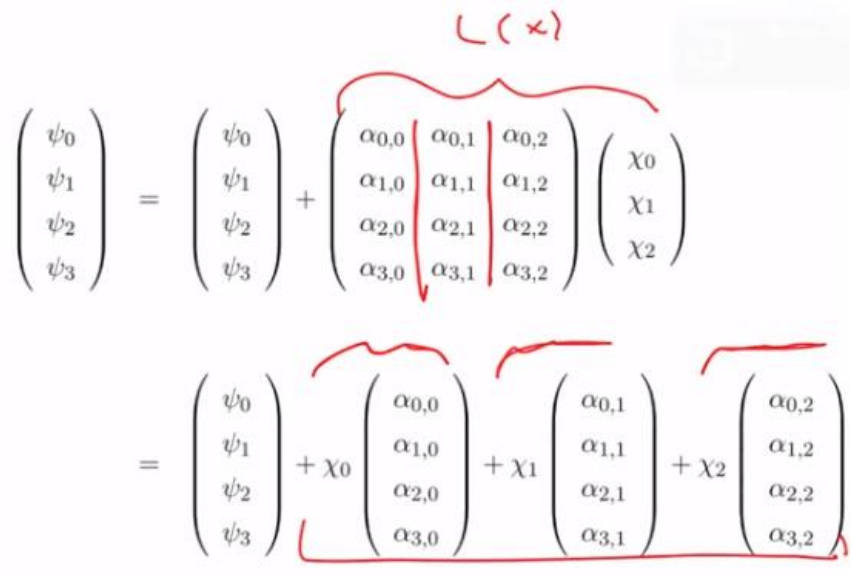
$$\underbrace{\begin{pmatrix} 0 & 0 \end{pmatrix}}_{u_{10}^T}\underbrace{\begin{pmatrix} 1 \\ 2 \end{pmatrix}}_{x_0} + \underbrace{(3)}_{\upsilon_{11}}\underbrace{(3)}_{\chi_1} + \underbrace{\begin{pmatrix} 1 & 2 \end{pmatrix}}_{u_{12}^T}\underbrace{\begin{pmatrix} 4 \\ 5 \end{pmatrix}}_{x_2} + \underbrace{0}_{\psi_1}$$

→ symmetric : a10^{T}을 a01^{T}로 대체하면 된다!



3.2.2. Matrix를 열로 나눠버려라!!

- linear combination으로 나타내는 방법도 있다. 이 방법은 좀 고민을 해봐야 한다. 왜냐하면 조금은 낯설기 때문이다. 경제수학 시간에 이것을 배웠었다. Ax를 나타낼 때, 세로로 scalar x column vector의 합으로 표현할 수 있다고.



Example

$$\begin{pmatrix} -1 & 0 & 2 \\ 2 & -1 & 1 \\ 3 & 1 & -1 \end{pmatrix} \begin{pmatrix} -1 \\ 2 \\ 1 \end{pmatrix} = (-1)\begin{pmatrix} -1 \\ 2 \\ 3 \end{pmatrix} + 2\begin{pmatrix} 0 \\ -1 \\ 1 \end{pmatrix} + 1\begin{pmatrix} -1 \\ 2 \\ 1 \end{pmatrix}$$

$$= \begin{pmatrix} (-1)(-1) \\ (-1)\ 2 \\ (-1)\ 3 \end{pmatrix} + \begin{pmatrix} 2\ (0) \\ 2\ (-1) \\ 2\ (1) \end{pmatrix} + \begin{pmatrix} 1\ (-1) \\ 1\ (2) \\ 1\ (1) \end{pmatrix}$$

- 알고리즘 :

$$\begin{pmatrix} \psi_0 \\ \psi_1 \\ \psi_2 \\ \psi_3 \end{pmatrix} = \begin{pmatrix} \psi_0 \\ \psi_1 \\ \psi_2 \\ \psi_3 \end{pmatrix} + \begin{pmatrix} \alpha_{0,0} & \alpha_{0,1} & \alpha_{0,2} \\ \alpha_{1,0} & \alpha_{1,1} & \alpha_{1,2} \\ \alpha_{2,0} & \alpha_{2,1} & \alpha_{2,2} \\ \alpha_{3,0} & \alpha_{3,1} & \alpha_{3,2} \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix}$$

$$= \begin{pmatrix} \psi_0 \\ \psi_1 \\ \psi_2 \\ \psi_3 \end{pmatrix} + \chi_0 \begin{pmatrix} \alpha_{0,0} \\ \alpha_{1,0} \\ \alpha_{2,0} \\ \alpha_{3,0} \end{pmatrix} + \chi_1 \begin{pmatrix} \alpha_{0,1} \\ \alpha_{1,1} \\ \alpha_{2,1} \\ \alpha_{3,1} \end{pmatrix} + \chi_2 \begin{pmatrix} \alpha_{0,2} \\ \alpha_{1,2} \\ \alpha_{2,2} \\ \alpha_{3,2} \end{pmatrix}$$

for $j = 0, \ldots, n-1$
  for $i = 0, \ldots, m-1$
    $\psi_i := \psi_i + \alpha_{i,j}\chi_j$
  endfor
endfor

→ 다시 표현을 하자면, axpy와 유사한 꼴이 된다. Y는 vector이다.

$$y = y + \left(\begin{array}{c|c|c} a_0 & a_1 & a_2 \end{array} \right) \begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix} = y + \chi_0 a_0 + \chi_1 a_1 + \chi_2 a_2$$

for $j = 0, \ldots, n-1$
  for $i = 0, \ldots, m-1$
    $\psi_i := \psi_i + \alpha_{i,j}\chi_j$
  endfor
endfor

$y := y + \chi_j a_j$
$y := \chi_j a_j + y$

axpy

→ 이런 식으로 한다.

y^{cur}	A	x	y^{next}
$\begin{pmatrix} -1 \\ -2 \\ 0 \\ -3 \\ 1 \end{pmatrix}$	$\left(\begin{array}{c\|cccc} 1 & -1 & 3 & 2 & -2 \\ 2 & -2 & 1 & 0 & -1 \\ 0 & -4 & 3 & 2 & 1 \\ 3 & 1 & -2 & 1 & 0 \\ -1 & 2 & 1 & -1 & -2 \end{array}\right)$	$\left(\begin{array}{c} -1 \\ \hline 0 \\ \hline 2 \\ -1 \\ 1 \end{array}\right)$	$\begin{pmatrix} -1 \\ -2 \\ 0 \\ -3 \\ 1 \end{pmatrix}$

$$y^{\text{next}} = (0)\begin{pmatrix} -1 \\ -2 \\ -4 \\ 1 \\ 2 \end{pmatrix} + \begin{pmatrix} -1 \\ -2 \\ 0 \\ -3 \\ 1 \end{pmatrix} = \begin{pmatrix} -1 \\ -2 \\ 0 \\ -3 \\ 1 \end{pmatrix}$$

→ column에다가 계속 추가한다.

→ 대각선으로 쭉 내려가면 어떻게 될까?

y^{cur}	A	x	y^{next}
$\left(\begin{array}{c} -1 \\ -2 \\ \hline 0 \\ \hline -3 \\ 1 \end{array}\right)$	$\left(\begin{array}{cc\|c\|cc} 1 & -1 & 3 & 2 & -2 \\ 2 & -2 & 1 & 0 & -1 \\ \hline 0 & -4 & 3 & 2 & 1 \\ \hline 3 & 1 & -2 & 1 & 0 \\ -1 & 2 & 1 & -1 & -2 \end{array}\right)$	$\left(\begin{array}{c} -1 \\ 0 \\ \hline 2 \\ \hline -1 \\ 1 \end{array}\right)$	$\left(\begin{array}{c} 5 \\ 0 \\ \hline 6 \\ \hline -7 \\ 3 \end{array}\right)$

$$y^{\text{next}} = (2)\left(\begin{array}{c} 3 \\ 1 \\ \hline 3 \\ \hline -2 \\ 1 \end{array}\right) + \left(\begin{array}{c} 1 \\ -2 \\ \hline 0 \\ \hline -3 \\ 1 \end{array}\right) = \left(\begin{array}{c} 5 \\ 0 \\ \hline 6 \\ \hline -7 \\ 3 \end{array}\right)$$

$$\left(\begin{array}{c} y_0^{\text{next}} \\ \hline \psi_1^{\text{next}} \\ \hline y_2^{\text{next}} \end{array}\right) = \left(\begin{array}{c} (2)\begin{pmatrix} 3 \\ 1 \end{pmatrix} \\ \hline (2) \times (3) \\ \hline (2)\begin{pmatrix} -2 \\ 1 \end{pmatrix} \end{array}\right) + \left(\begin{array}{c} \begin{pmatrix} 1 \\ -2 \end{pmatrix} \\ \hline 0 \\ \hline \begin{pmatrix} -3 \\ 1 \end{pmatrix} \end{array}\right) = \left(\begin{array}{c} \begin{pmatrix} 5 \\ 0 \end{pmatrix} \\ \hline 6 \\ \hline \begin{pmatrix} -7 \\ 3 \end{pmatrix} \end{array}\right)$$

$$\begin{aligned} y_0^{\text{next}} &:= (2)\begin{pmatrix} 3 \\ 1 \end{pmatrix} + \begin{pmatrix} 1 \\ -2 \end{pmatrix} = \chi_1 a_{01} + y_0^{\text{cur}} \\ \psi_1^{\text{next}} &:= (2) \times (3) + 0 = \chi_1 \alpha_{11} + \psi_1^{\text{cur}} \\ y_2^{\text{next}} &:= (2)\begin{pmatrix} -2 \\ 1 \end{pmatrix} + \begin{pmatrix} -3 \\ 1 \end{pmatrix} = \chi_1 a_{21} + y_2^{\text{cur}} \end{aligned}$$

→ 대각선 가운데를 기준으로 해서 column 위 아래, 이렇게 3파트로 나눠져서 axpy계산을 하게 된다. Lower triangular할 때 이 계산이 용이하다.

Why is this alternative way important?

y^{cur}	A	x	y^{next}
$\left(\begin{array}{r} -1 \\ -2 \\ \hline 0 \\ -3 \\ 1 \end{array}\right)$	$\left(\begin{array}{rr\|r\|rr} 1 & 0 & 0 & 0 & 0 \\ 2 & -2 & 0 & 0 & 0 \\ \hline 0 & -4 & 3 & 0 & 0 \\ \hline 3 & 1 & -2 & 1 & 0 \\ -1 & 2 & 1 & -1 & -2 \end{array}\right)$	$\left(\begin{array}{r} -1 \\ 0 \\ \hline 2 \\ -1 \\ 1 \end{array}\right)$	$\left(\begin{array}{r} -1 \\ -2 \\ \hline 6 \\ -7 \\ 3 \end{array}\right)$

$$
\begin{array}{rcll}
y_0^{\text{next}} & := & (2)\begin{pmatrix} 0 \\ 0 \end{pmatrix} + \begin{pmatrix} 1 \\ -2 \end{pmatrix} & = \chi_1 a_{01} + y_0^{\text{cur}} \\
\psi_1^{\text{next}} & := & (2) \times (3) + 0 & = \chi_1 \alpha_{11} + \psi_1^{\text{cur}} \\
y_2^{\text{next}} & := & (2)\begin{pmatrix} -2 \\ 1 \end{pmatrix} + \begin{pmatrix} -3 \\ 1 \end{pmatrix} & = \chi_1 a_{21} + y_2^{\text{cur}}
\end{array}
$$

→ triangular!!

(Assume that initially $y = 0$.)

$$
\begin{pmatrix} y_0 \\ \hline \psi_1 \\ \hline y_2 \end{pmatrix} := \left(\begin{array}{rr|r|rr} -1 & 2 & 4 & 1 & 0 \\ 0 & 0 & -1 & -2 & 1 \\ \hline 0 & 0 & 3 & 1 & 2 \\ \hline 0 & 0 & 0 & 4 & 3 \\ 0 & 0 & 0 & 0 & 2 \end{array}\right) \begin{pmatrix} 1 \\ 2 \\ \hline 3 \\ \hline 4 \\ 5 \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ \hline 0 \\ \hline 0 \\ 0 \end{pmatrix}
$$

$$
\begin{pmatrix} y_0 \\ \hline \psi_1 \\ \hline y_2 \end{pmatrix} := \chi_1 \begin{pmatrix} u_{01} \\ \hline \upsilon_{11} \\ \hline u_{21} \end{pmatrix} + \begin{pmatrix} y_0 \\ \hline \psi_1 \\ \hline y_2 \end{pmatrix}
$$

$$
\begin{array}{rcl}
y_0 & := & \chi_1 u_{01} + y_0 \\
\psi_1 & := & \chi_1 \upsilon_{11} + \psi_1 \\
y_2 & := & \chi_1 u_{21} + y_2
\end{array}
$$

→ symmetric :

(Assume that initially $y = 0$.)

$$
\begin{pmatrix} y_0 \\ \hline \psi_1 \\ \hline y_2 \end{pmatrix} := \left(\begin{array}{rr|r|rr} -1 & 2 & 4 & 1 & 0 \\ 2 & 0 & -1 & -2 & 1 \\ \hline 4 & -1 & 3 & 1 & 2 \\ \hline 1 & -2 & 1 & 4 & 3 \\ 0 & 1 & 2 & 3 & 2 \end{array}\right) \begin{pmatrix} 1 \\ 2 \\ \hline 3 \\ \hline 4 \\ 5 \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ \hline 0 \\ \hline 0 \\ 0 \end{pmatrix}
$$

$$
\begin{pmatrix} y_0 \\ \hline \psi_1 \\ \hline y_2 \end{pmatrix} := \chi_1 \begin{pmatrix} a_{01} \\ \hline \alpha_{11} \\ \hline a_{21} \end{pmatrix} + \begin{pmatrix} y_0 \\ \hline \psi_1 \\ \hline y_2 \end{pmatrix}
$$

$$
\begin{array}{rcl}
y_0 & := & \chi_1 a_{01} + y_0 \\
\psi_1 & := & \chi_1 \alpha_{11} + \psi_1 \\
y_2 & := & \chi_1 a_{21} + y_2
\end{array}
$$

복습)

$$
\begin{aligned}
y &= L(x) && < \text{Reason} > \\
&= L\left(\textstyle\sum_{j=0}^{n-1} \chi_j e_j.\right) && < x = \textstyle\sum_{j=0}^{n-1} \chi_j e_j > \\
&= \textstyle\sum_{j=0}^{n-1} \chi_j L(e_j) && < L(\textstyle\sum_{j=0}^{n-1} \alpha_j v_j) = \textstyle\sum_{j=0}^{n-1} \alpha_j L(v_j) > \\
&= \textstyle\sum_{j=0}^{n-1} \chi_j \underbrace{L(e_j)}_{a_j} && < \text{Let } a_j = L(e_j) > \\
&= \textstyle\sum_{j=0}^{n-1} \chi_j a_j.
\end{aligned}
$$

alternatively

Less concisely:

$$
\begin{aligned}
L(x) &= L(\chi_0 e_0 + \chi_1 e_1 + \cdots + \chi_{n-1} e_{n-1}) \\
&= \chi_0 a_0 + \chi_1 a_1 + \cdots + \chi_{n-1} a_{n-1},
\end{aligned}
$$

where $a_j = L(e_j)$.

3.3. Partition and transposing in matrix-vector multiplication

3.3.1. Partitioned matrix-vector multiplication

$$
\begin{aligned}
y &= \left(\begin{array}{c} y_0 \\ \hline \psi_1 \\ \hline y_2 \end{array}\right) = \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right) \left(\begin{array}{c} x_0 \\ \hline \chi_1 \\ \hline x_2 \end{array}\right) \\
&= \left(\begin{array}{c} A_{00}x_0 + a_{01}\chi_1 + A_{02}x_2 \\ \hline a_{10}^T x_0 + \alpha_{11}\chi_1 + a_{12}^T x_2 \\ \hline A_{20}x_0 + a_{21}\chi_1 + A_{22}x_2 \end{array}\right)
\end{aligned}
$$

$$= \left(\begin{array}{c} \left(\begin{array}{cc} -1 & 2 \\ 1 & 0 \end{array}\right)\left(\begin{array}{c} 1 \\ 2 \end{array}\right) + \left(\begin{array}{c} 4 \\ -1 \end{array}\right)3 + \left(\begin{array}{cc} 1 & 0 \\ -2 & 1 \end{array}\right)\left(\begin{array}{c} 4 \\ 5 \end{array}\right) \\ \hline \left(\begin{array}{cc} 2 & -1 \end{array}\right)\left(\begin{array}{c} 1 \\ 2 \end{array}\right) + \left(\begin{array}{c} 3 \end{array}\right)3 + \left(\begin{array}{cc} 1 & 2 \end{array}\right)\left(\begin{array}{c} 4 \\ 5 \end{array}\right) \\ \hline \left(\begin{array}{cc} 1 & 2 \\ -1 & -2 \end{array}\right)\left(\begin{array}{c} 1 \\ 2 \end{array}\right) + \left(\begin{array}{c} 3 \\ 0 \end{array}\right)3 + \left(\begin{array}{cc} 4 & 3 \\ 1 & 2 \end{array}\right)\left(\begin{array}{c} 4 \\ 5 \end{array}\right) \end{array}\right)$$

$$= \left(\begin{array}{c} \left(\begin{array}{c} (-1)\times(1)+(2)\times(2) \\ (1)\times(1)+(0)\times(2) \end{array}\right) + \left(\begin{array}{c} (4)\times(3) \\ (-1)\times(3) \end{array}\right) + \left(\begin{array}{c} (1)\times(4)+(0)\times(5) \\ (-2)\times(4)+(1)\times(5) \end{array}\right) \\ \hline (2)\times(1)+(-1)\times(2) + \quad (3)\times(3) + \quad (1)\times(4)+(2)\times(5) \\ \hline \left(\begin{array}{c} (1)\times(1)+(2)\times(2) \\ (-1)\times(1)+(-2)\times(2) \end{array}\right) + \left(\begin{array}{c} (3)\times 3 \\ (0)\times 3 \end{array}\right) + \left(\begin{array}{c} (4)\times(4)+(3)\times(5) \\ (1)\times(4)+(2)\times(5) \end{array}\right) \end{array}\right)$$

$$\left(\begin{array}{c} y_0 \\ \hline y_1 \\ \hline \vdots \\ \hline y_{M-1} \end{array}\right) = \left(\begin{array}{c|c|c|c} A_{0,0} & A_{0,1} & \cdots & A_{0,N-1} \\ \hline A_{1,0} & A_{1,1} & \cdots & A_{1,N-1} \\ \hline \vdots & \vdots & \ddots & \vdots \\ \hline A_{M-1,0} & A_{M-1,1} & \cdots & A_{M-1,N-1} \end{array}\right)\left(\begin{array}{c} x_0 \\ \hline x_1 \\ \hline \vdots \\ \hline x_{N-1} \end{array}\right)$$

$$= \left(\begin{array}{c} A_{0,0}x_0 + A_{0,1}x_1 + \cdots + A_{0,N-1}x_{N-1} \\ \hline A_{1,0}x_0 + A_{1,1}x_1 + \cdots + A_{1,N-1}x_{N-1} \\ \hline \vdots \\ \hline A_{M-1,0}x_0 + A_{M-1,1}x_1 + \cdots + A_{M-1,N-1}x_{N-1} \end{array}\right)$$

In other words...

$$y_i = \sum_{j=0}^{N-1} A_{i,j}x_j.$$

3.3.2. Transposing a partitioned matrix

Let $A \in \mathbb{R}^{m\times n}$ be partitioned as follows:

$$A = \left(\begin{array}{c|c|c|c} A_{0,0} & A_{0,1} & \cdots & A_{0,N-1} \\ \hline A_{1,0} & A_{1,1} & \cdots & A_{1,N-1} \\ \hline \vdots & \vdots & & \vdots \\ \hline A_{M-1,0} & A_{M-1,1} & \cdots & A_{M-1,N-1} \end{array}\right),$$

where $A_{i,j} \in \mathbb{R}^{m_i\times n_j}$. Then

$$A^T = \left(\begin{array}{c|c|c|c} A_{0,0}^T & A_{1,0}^T & \cdots & A_{M-1,0}^T \\ \hline A_{0,1}^T & A_{1,1}^T & \cdots & A_{M-1,1}^T \\ \hline \vdots & \vdots & & \vdots \\ \hline A_{0,N-1}^T & A_{1,N-1}^T & \cdots & A_{M-1,N-1}^T \end{array}\right).$$

- 전체 matrix를 transpose하면 안의 submatrix를 각각 transpose한 것과 같다.

Look at the different submatrices:

$$A = \left(\begin{array}{cc|cc} 1 & -1 & 3 & 2 \\ 2 & -2 & 1 & 0 \\ \hline 0 & -4 & 3 & 2 \end{array}\right), \quad A^T = \left(\begin{array}{cc|c} 1 & 2 & 0 \\ -1 & -2 & -4 \\ \hline 3 & 1 & 3 \\ 2 & 0 & 2 \end{array}\right).$$

$$A = \left(\begin{array}{cc|cc} 1 & -1 & 3 & 2 \\ 2 & -2 & 1 & 0 \\ \hline 0 & -4 & 3 & 2 \end{array}\right), \quad A^T = \left(\begin{array}{c|c} \begin{pmatrix} 1 & -1 \\ 2 & -2 \end{pmatrix}^T & \begin{pmatrix} 0 & -4 \end{pmatrix}^T \\ \hline \begin{pmatrix} 3 & 2 \\ 1 & 0 \end{pmatrix}^T & \begin{pmatrix} 3 & 2 \end{pmatrix}^T \end{array}\right)$$

- row로 partition된 것 transpose하기

Special case: matrix partitioned by rows

Let $A = \left(\begin{array}{c} \widetilde{a}_0^T \\ \hline \widetilde{a}_1^T \\ \hline \vdots \\ \hline \widetilde{a}_{m-1}^T \end{array}\right)$ where each $\widetilde{a}_i^T$ is a row of A, then

$$A^T = \left(\begin{array}{c} \widetilde{a}_0^T \\ \hline \widetilde{a}_1^T \\ \hline \vdots \\ \hline \widetilde{a}_{m-1}^T \end{array}\right)^T = \left((\widetilde{a}_0^T)^T \,\middle|\, (\widetilde{a}_1^T)^T \,\middle|\, \cdots \,\middle|\, (\widetilde{a}_{m-1}^T)^T \right)$$
$$= \left(\widetilde{a}_0 \,\middle|\, \widetilde{a}_1 \,\middle|\, \cdots \,\middle|\, \widetilde{a}_{m-1} \right).$$

Rows of A become columns of A^T

- column으로 partition된 것의 transpose

Special case: matrix partitioned by columns

If

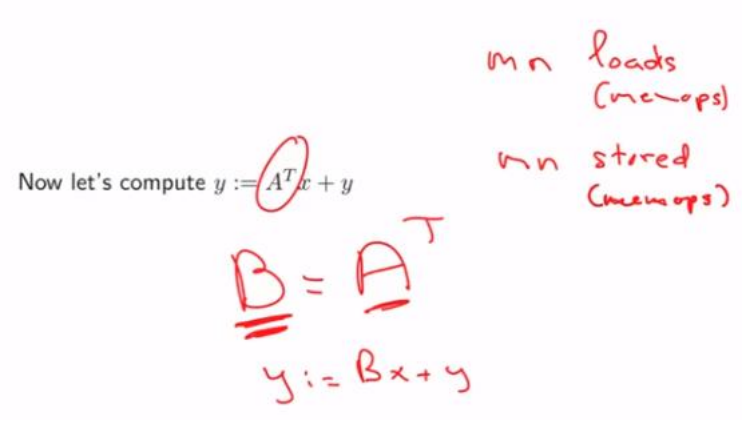
$$A = \left(a_0 \,\middle|\, a_1 \,\middle|\, \cdots \,\middle|\, a_{n-1} \right),$$

where each a_j is a column of A, then

$$A^T = \left(a_0 \,\middle|\, a_1 \,\middle|\, \cdots \,\middle|\, a_{n-1} \right)^T = \left(\begin{array}{c} a_0^T \\ \hline a_1^T \\ \hline \vdots \\ \hline a_{m-1}^T \end{array}\right).$$

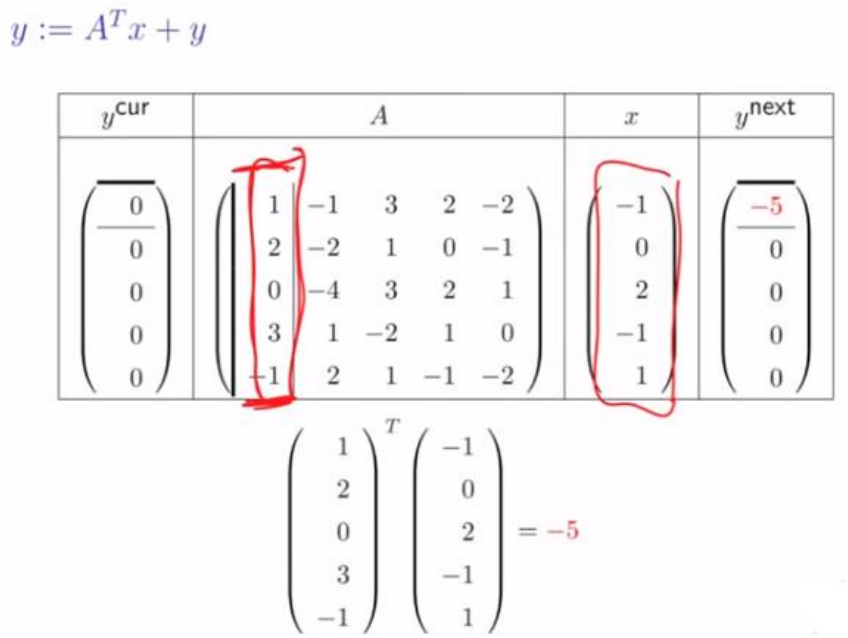
Columns of A become rows of A^T

3.3.3. Transpose Matrix-vector multiplication



→ B로 치환해서 하고 싶지만 memory operation이 거슬린다.

- 그러면 dot product 형태로 연산을 한다고 할 때, row 대신에 column을 하면 transpose 행렬을 구하지 않고도 할 수 있지 않는가!! 천재다. 쉽게 가능하다. No need to explicitly transpose matrix.



4. Matrix-Matrix multiplication

- markov process : a stochastic model describing a sequence of possible events in which the probability of each event depends only on the state attained in the previous event.

Preparation

4.2.1. Partitioned Matrix-Vector Multiplication

4.2.2. Transposing a Partitioned Matrix

4.2.3. Matrix-Vector Multiplication, Again

Matrix-Vector Multiplication with Special Matrices

4.3.1. Transpose Matrix-Vector Multiplication

4.3.2. Triangular Matrix-Vector Multiplication

4.3.3. Symmetric Matrix-Vector Multiplication

Matrix-Matrix Multiplication (Product)

4.4.1. Motivation

4.4.2. From Composing Linear Transformations to Matrix-Matrix Multiplication

4.4.3. Computing the Matrix-Matrix Product

4.4.4. Special Shapes

4.4.5. Cost

Enrichment

4.5.1. Hidden Markov Processes

4.1. Matrix-Matrix multiplication meaning : 행렬과 행렬을 곱한다는 것이 무슨 뜻인가?

- Matrix-vector multiplication → used to compactly write a set of simultaneous linear equations

Notation:

- $\chi_s^{(k)}$: probability that it will be sunny k days from now.
- $\chi_c^{(k)}$: probability that it will be cloudy k days from now.
- $\chi_r^{(k)}$: probability that it will be rainy k days from now.

		Today		
		sunny	cloudy	rainy
Tomorrow	sunny	0.4	0.3	0.1
	cloudy	0.4	0.3	0.6
	rainy	0.2	0.4	0.3

$$
\begin{array}{rcl}
\chi_s^{(k+1)} & = & 0.4 \times \chi_s^{(k)} + 0.3 \times \chi_c^{(k)} + 0.1 \times \chi_r^{(k)} \\
\chi_c^{(k+1)} & = & 0.4 \times \chi_s^{(k)} + 0.3 \times \chi_c^{(k)} + 0.6 \times \chi_r^{(k)} \\
\chi_r^{(k+1)} & = & 0.2 \times \chi_s^{(k)} + 0.4 \times \chi_c^{(k)} + 0.3 \times \chi_r^{(k)}
\end{array}
$$

$$
\begin{pmatrix} \chi_s^{(k+1)} \\ \chi_c^{(k+1)} \\ \chi_r^{(k+1)} \end{pmatrix} = \begin{pmatrix} 0.4 & 0.3 & 0.1 \\ 0.4 & 0.3 & 0.6 \\ 0.2 & 0.4 & 0.3 \end{pmatrix} \begin{pmatrix} \chi_s^{(k)} \\ \chi_c^{(k)} \\ \chi_r^{(k)} \end{pmatrix},
$$

or

$$
x^{(k+1)} = P x^{(k)}.
$$

- (State) vector x
- (Transition) matrix P

- 내일의 날씨!

The vector of probabilities for tomorrow's weather, $x^{(1)}$, is given by

$$\begin{pmatrix} \chi_s^{(1)} \\ \chi_c^{(1)} \\ \chi_r^{(1)} \end{pmatrix} = \begin{pmatrix} 0.4 & 0.3 & 0.1 \\ 0.4 & 0.3 & 0.6 \\ 0.2 & 0.4 & 0.3 \end{pmatrix} \begin{pmatrix} \chi_s^{(0)} \\ \chi_c^{(0)} \\ \chi_r^{(0)} \end{pmatrix}$$

$$= \begin{pmatrix} 0.4 & 0.3 & 0.1 \\ 0.4 & 0.3 & 0.6 \\ 0.2 & 0.4 & 0.3 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}$$

$$= \begin{pmatrix} 0.4 \times 0 + 0.3 \times 1 + 0.1 \times 0 \\ 0.4 \times 0 + 0.3 \times 1 + 0.6 \times 0 \\ 0.2 \times 0 + 0.4 \times 1 + 0.3 \times 0 \end{pmatrix} = \begin{pmatrix} 0.3 \\ 0.3 \\ 0.4 \end{pmatrix}.$$

Ah! You should not be surprised! $Pe_1 = p_1$!!

- 모래의 날씨를 구하려면 거기다가 다시 한번 곱하면 된다!!

If today is cloudy, the vector of probabilities for the day after tomorrow, $x^{(2)}$, is given by

$$\begin{pmatrix} \chi_s^{(2)} \\ \chi_c^{(2)} \\ \chi_r^{(2)} \end{pmatrix} = \begin{pmatrix} 0.4 & 0.3 & 0.1 \\ 0.4 & 0.3 & 0.6 \\ 0.2 & 0.4 & 0.3 \end{pmatrix} \begin{pmatrix} \chi_s^{(1)} \\ \chi_c^{(1)} \\ \chi_r^{(1)} \end{pmatrix}$$

$$= \begin{pmatrix} 0.4 & 0.3 & 0.1 \\ 0.4 & 0.3 & 0.6 \\ 0.2 & 0.4 & 0.3 \end{pmatrix} \begin{pmatrix} 0.3 \\ 0.3 \\ 0.4 \end{pmatrix}$$

$$= \begin{pmatrix} 0.4 \times 0.3 + 0.3 \times 0.3 + 0.1 \times 0.4 \\ 0.4 \times 0.3 + 0.3 \times 0.3 + 0.6 \times 0.4 \\ 0.2 \times 0.3 + 0.4 \times 0.3 + 0.3 \times 0.4 \end{pmatrix}$$

$$= \begin{pmatrix} 0.25 \\ 0.45 \\ 0.30 \end{pmatrix}.$$

$$\begin{pmatrix} \chi_s^{(2)} \\ \chi_c^{(2)} \\ \chi_r^{(2)} \end{pmatrix} = \begin{pmatrix} 0.4 & 0.3 & 0.1 \\ 0.4 & 0.3 & 0.6 \\ 0.2 & 0.4 & 0.3 \end{pmatrix} \begin{pmatrix} \chi_s^{(1)} \\ \chi_c^{(1)} \\ \chi_r^{(1)} \end{pmatrix}$$

$$= \underbrace{\begin{pmatrix} 0.4 & 0.3 & 0.1 \\ 0.4 & 0.3 & 0.6 \\ 0.2 & 0.4 & 0.3 \end{pmatrix} \underbrace{\left(\begin{pmatrix} 0.4 & 0.3 & 0.1 \\ 0.4 & 0.3 & 0.6 \\ 0.2 & 0.4 & 0.3 \end{pmatrix} \begin{pmatrix} \chi_s^{(0)} \\ \chi_c^{(0)} \\ \chi_r^{(0)} \end{pmatrix} \right)}_{L_P(x)}}_{L_P(L_P(x))}$$

$$= \underbrace{Q \begin{pmatrix} \chi_s^{(0)} \\ \chi_c^{(0)} \\ \chi_r^{(0)} \end{pmatrix}}_{L_Q(x)},$$

→ Q행렬이 P행렬 * P행렬

- Matlab에서 하려면

For I = 1:7

i

x = P * x

end

- 예시2 : rotation을 두 번하면?

$$
\begin{aligned}
&\left(\begin{array}{c|c} \cos(\theta+\rho) & -\sin(\theta+\rho) \\ \hline \sin(\theta+\rho) & \cos(\theta+\rho) \end{array}\right)\begin{pmatrix}\chi_0\\ \chi_1\end{pmatrix} \\
&= R_{\theta+\rho}(\begin{pmatrix}\chi_0\\ \chi_1\end{pmatrix}) \\
&= R_{\rho}(R_{\theta}(\begin{pmatrix}\chi_0\\ \chi_1\end{pmatrix})) \\
&= \left(\begin{array}{c|c} \cos(\rho) & -\sin(\rho) \\ \hline \sin(\rho) & \cos(\rho) \end{array}\right)\left(\left(\begin{array}{c|c} \cos(\theta) & -\sin(\theta) \\ \hline \sin(\theta) & \cos(\theta) \end{array}\right)\begin{pmatrix}\chi_0\\ \chi_1\end{pmatrix}\right) \\
&= \left(\begin{array}{c|c} \cos(\rho) & -\sin(\rho) \\ \hline \sin(\rho) & \cos(\rho) \end{array}\right)\left(\begin{array}{c|c} \cos(\theta) & -\sin(\theta) \\ \hline \sin(\theta) & \cos(\theta) \end{array}\right)\begin{pmatrix}\chi_0\\ \chi_1\end{pmatrix} \\
&= \left(\begin{array}{c|c} \cos(\rho)\cos(\theta)-\sin(\rho)\sin(\theta) & -\cos(\rho)\sin(\theta)-\sin(\rho)\cos(\theta) \\ \hline \cos(\rho)\sin(\theta)+\sin(\rho)\cos(\theta) & \cos(\rho)\cos(\theta)-\sin(\rho)\sin(\theta) \end{array}\right)\begin{pmatrix}\chi_0\\ \chi_1\end{pmatrix}
\end{aligned}
$$

$$
\begin{aligned}
\cos(\theta+\rho) &= \cos(\rho)\cos(\theta)-\sin(\rho)\sin(\theta) \\
\sin(\theta+\rho) &= \cos(\rho)\sin(\theta)+\sin(\rho)\cos(\theta)
\end{aligned}
$$

You learned these as the law of cosines and law of sines in a geometry class!

- linear transformation이라고 가정하면 Q라는 행렬이 정말로 존재를 하고, L(x)에 unit basis vector를 넣음으로써 무엇인지 알 수 있게 된다.

Summary

- Is $L_Q = L_P(L_P(x))$ a linear transformation?
- Is $L_C = L_A(L_B(x))$ a linear transformation?
- If it is, how are matrices A, B, and C related?

4.2. Why!!! Not HOW!! Multiplication of matrices

- $L_C(x) = L_A(L_B(x))$라고 할 때, $L_C(x)$가 linear transformation인가? Yes.

Let $L_A : \mathbb{R}^k \to \mathbb{R}^m$ and $L_B : \mathbb{R}^n \to \mathbb{R}^k$ both be linear transformations and, for all $x \in \mathbb{R}^n$, define the function $L_C : \mathbb{R}^n \to \mathbb{R}^m$ by $L_C(x) = L_A(L_B(x))$. $L_C(x)$ is a linear transformations.

Always/Sometimes/Never

증명) 증명을 하려면, 두 가지를 모두 증명해야 한다. → 아하 linear transformation이구나

Always

Let $x, y \in \mathbb{R}^n$ and $\alpha \in \mathbb{R}$.

- $L_C(\alpha x) = \alpha L_C(x)$?
- $L_C(x + y) = L_C(x) + L_C(y)$?.

Always

Let $x, y \in \mathbb{R}^n$ and $\alpha \in \mathbb{R}$.

- $L_C(\alpha x) = L_A(L_B(\alpha x)) = L_A(\alpha L_B(x)) = \alpha L_A(L_B(x)) = \alpha L_C(x)$.
- $L_C(x + y) = L_A(L_B(x + y)) = L_A(L_B(x) + L_B(y)) = L_A(L_B(x)) + L_A(L_B(y)) = L_C(x) + L_C(y)$.

- 어떻게 계산하는가? → matrix-vector multiplication이다.

How to compute $C = AB$?

- $C = \left(c_0 \mid c_1 \mid \cdots \mid c_{n-1} \right)$ and $B = \left(b_0 \mid b_1 \mid \cdots \mid b_{n-1} \right)$.

$$c_j = Ce_j = L_C(e_j) = L_A(L_B(e_j)) = A(Be_j) = Ab_j$$

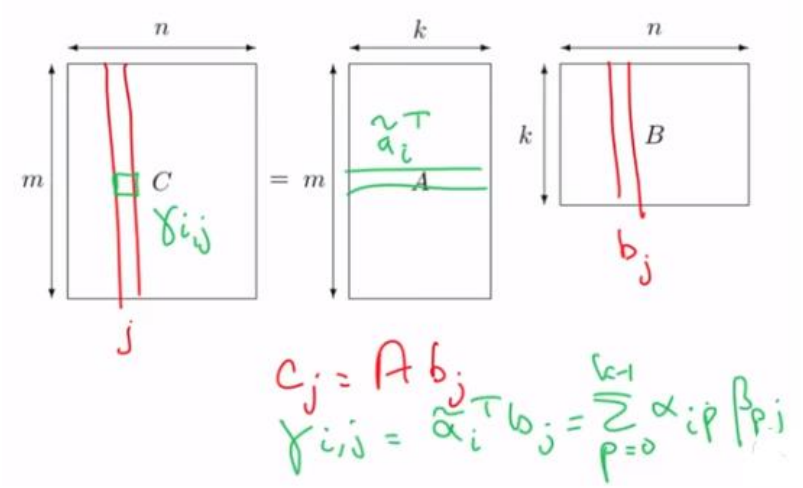
$$\begin{pmatrix} \gamma_{0,j} \\ \gamma_{1,j} \\ \vdots \\ \gamma_{m-1,j} \end{pmatrix} = \begin{pmatrix} \tilde{a}_0^T \\ \tilde{a}_1^T \\ \vdots \\ \tilde{a}_{m-1}^T \end{pmatrix} b_j = \begin{pmatrix} \tilde{a}_0^T b_j \\ \tilde{a}_1^T b_j \\ \vdots \\ \tilde{a}_{m-1}^T b_j \end{pmatrix}$$

c_j

- $\gamma_{i,j} = \tilde{a}_i^T b_j = \begin{pmatrix} \alpha_{i,0} & \alpha_{i,1} & \cdots & \alpha_{i,k-1} \end{pmatrix} \begin{pmatrix} \beta_{0,j} \\ \beta_{1,j} \\ \vdots \\ \beta_{k-1,j} \end{pmatrix} =$

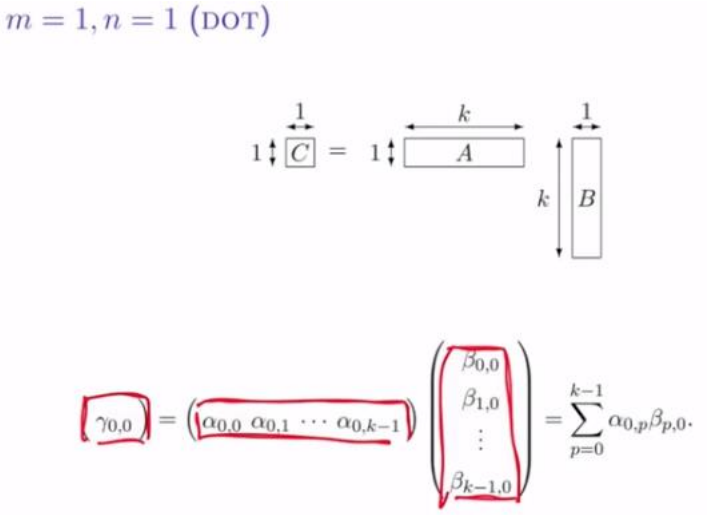
$$\alpha_{i,0}\beta_{0,j} + \alpha_{i,1}\beta_{1,j} + \cdots + \alpha_{i,k-1}\beta_{k-1,j} = \sum_{p=0}^{k-1} \alpha_{i,p}\beta_{p,j}.$$

→ C_j의 각 감마 component는 dot product임을 알 수 있다.

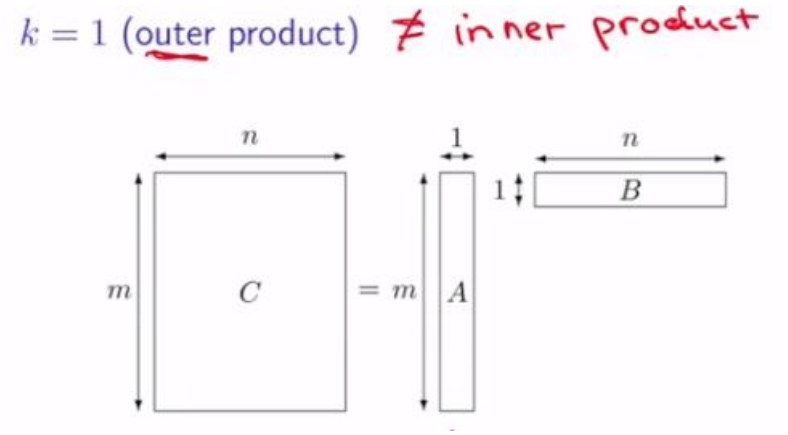


4.3. 다양한 경우와 예시

- inner product : dot product



- outer product :



→ 결론 : matrix-vector operation은 matrix-matrix multiplication의 특수한 경우이다.

- 사용하는 경우

(1) Eugeny Onegin

(2) Shannon's application to information theory : A mathematical theory of communication.

(3) Scherr's application to computer performance evaluation.

(4) Baum's application to hidden markov models.

(5) S. Brin and Larry page application to web search(PageRank)

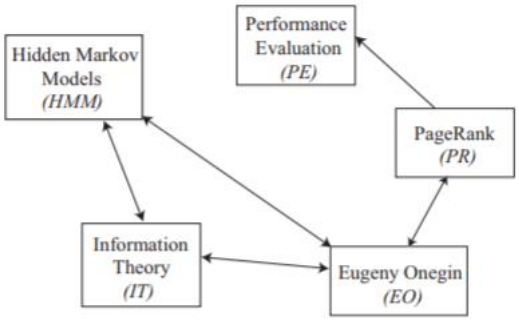


FIG. 7.1. *Graph of relationships between five Markov applications*

5단원

Observations

5.2.1. Partitioned Matrix-Matrix Multiplication

5.2.2. Properties

5.2.3. Transposing a Product of Matrices

5.2.4. Matrix-Matrix Multiplication with Special Matrices

Algorithms for Computing Matrix-Matrix Multiplication

5.3.1. Lots of Loops

5.3.2. Matrix-Matrix Multiplication by Columns

5.3.3. Matrix-Matrix Multiplication by Rows

5.3.4. Matrix-Matrix Multiplication with **Rank-1 Updates**

Enrichment

5.4.1. Slicing and Dicing for Performance

5.1. Partitioned Matrix-Matrix multiplication

* 주의 : 행렬간 곱은 commute가 안 된다. (교환법칙이 성립하지 않는다.)

(1) C=AB을 할 때, column으로 partition을 하고 싶을 때

→ A가 쳐들어간다. B의 각각의 column이랑 같이 곱한다고 생각하면 된다.

$$\left(C_0 \mid C_1 \right) = A \left(B_0 \mid B_1 \right) = \left(AB_0 \mid AB_1 \right)$$

$$\left(\begin{array}{cc|cc} \gamma_{0,0} & \gamma_{0,1} & \gamma_{0,2} & \gamma_{0,3} \\ \gamma_{1,0} & \gamma_{1,1} & \gamma_{1,2} & \gamma_{1,3} \\ \gamma_{2,0} & \gamma_{2,1} & \gamma_{2,2} & \gamma_{2,3} \\ \gamma_{3,0} & \gamma_{3,1} & \gamma_{3,2} & \gamma_{3,3} \end{array}\right) =$$

$$\begin{pmatrix} \alpha_{0,0} & \alpha_{0,1} & \alpha_{0,2} & \alpha_{0,3} \\ \alpha_{1,0} & \alpha_{1,1} & \alpha_{1,2} & \alpha_{1,3} \\ \alpha_{2,0} & \alpha_{2,1} & \alpha_{2,2} & \alpha_{2,3} \\ \alpha_{3,0} & \alpha_{3,1} & \alpha_{3,2} & \alpha_{3,3} \end{pmatrix} \begin{pmatrix} \beta_{0,0} & \beta_{0,1} & \beta_{0,2} & \beta_{0,3} \\ \beta_{1,0} & \beta_{1,1} & \beta_{1,2} & \beta_{1,3} \\ \beta_{2,0} & \beta_{2,1} & \beta_{2,2} & \beta_{2,3} \\ \beta_{3,0} & \beta_{3,1} & \beta_{3,2} & \beta_{3,3} \end{pmatrix} =$$

$$\left(\begin{array}{cc|cc} \begin{pmatrix} \alpha_{0,0} & \alpha_{0,1} & \alpha_{0,2} & \alpha_{0,3} \\ \alpha_{1,0} & \alpha_{1,1} & \alpha_{1,2} & \alpha_{1,3} \\ \alpha_{2,0} & \alpha_{2,1} & \alpha_{2,2} & \alpha_{2,3} \\ \alpha_{3,0} & \alpha_{3,1} & \alpha_{3,2} & \alpha_{3,3} \end{pmatrix} & \begin{pmatrix} \beta_{0,0} & \beta_{0,1} \\ \beta_{1,0} & \beta_{1,1} \\ \beta_{2,0} & \beta_{2,1} \\ \beta_{3,0} & \beta_{3,1} \end{pmatrix} & \begin{pmatrix} \alpha_{0,0} & \alpha_{0,1} & \alpha_{0,2} & \alpha_{0,3} \\ \alpha_{1,0} & \alpha_{1,1} & \alpha_{1,2} & \alpha_{1,3} \\ \alpha_{2,0} & \alpha_{2,1} & \alpha_{2,2} & \alpha_{2,3} \\ \alpha_{3,0} & \alpha_{3,1} & \alpha_{3,2} & \alpha_{3,3} \end{pmatrix} & \begin{pmatrix} \beta_{0,2} & \beta_{0,3} \\ \beta_{1,2} & \beta_{1,3} \\ \beta_{2,2} & \beta_{2,3} \\ \beta_{3,2} & \beta_{3,3} \end{pmatrix} \end{array}\right)$$

(2) C=AB을 할 때, row로 partition을 하고 싶을 때

→ A를 row로 쪼갠다. B를 쪼개면 연산 자체가 안 된다.

$$\left(\frac{C_0}{C_1} \right) = \left(\frac{A_0}{A_1} \right) B = \left(\frac{A_0 B}{A_1 B} \right)$$

$$\left(\begin{array}{cccc} \gamma_{0,0} & \gamma_{0,1} & \gamma_{0,2} & \gamma_{0,3} \\ \gamma_{1,0} & \gamma_{1,1} & \gamma_{1,2} & \gamma_{1,3} \\ \hline \gamma_{2,0} & \gamma_{2,1} & \gamma_{2,2} & \gamma_{2,3} \\ \gamma_{3,0} & \gamma_{3,1} & \gamma_{3,2} & \gamma_{3,3} \end{array}\right) = \begin{pmatrix} \alpha_{0,0} & \alpha_{0,1} & \alpha_{0,2} & \alpha_{0,3} \\ \alpha_{1,0} & \alpha_{1,1} & \alpha_{1,2} & \alpha_{1,3} \\ \alpha_{2,0} & \alpha_{2,1} & \alpha_{2,2} & \alpha_{2,3} \\ \alpha_{3,0} & \alpha_{3,1} & \alpha_{3,2} & \alpha_{3,3} \end{pmatrix} \begin{pmatrix} \beta_{0,0} & \beta_{0,1} & \beta_{0,2} & \beta_{0,3} \\ \beta_{1,0} & \beta_{1,1} & \beta_{1,2} & \beta_{1,3} \\ \beta_{2,0} & \beta_{2,1} & \beta_{2,2} & \beta_{2,3} \\ \beta_{3,0} & \beta_{3,1} & \beta_{3,2} & \beta_{3,3} \end{pmatrix} =$$

$$\left(\begin{array}{c} \begin{pmatrix} \alpha_{0,0} & \alpha_{0,1} & \alpha_{0,2} & \alpha_{0,3} \\ \alpha_{1,0} & \alpha_{1,1} & \alpha_{1,2} & \alpha_{1,3} \end{pmatrix} \begin{pmatrix} \beta_{0,0} & \beta_{0,1} & \beta_{0,2} & \beta_{0,3} \\ \beta_{1,0} & \beta_{1,1} & \beta_{1,2} & \beta_{1,3} \\ \beta_{2,0} & \beta_{2,1} & \beta_{2,2} & \beta_{2,3} \\ \beta_{3,0} & \beta_{3,1} & \beta_{3,2} & \beta_{3,3} \end{pmatrix} \\ \hline \begin{pmatrix} \alpha_{2,0} & \alpha_{2,1} & \alpha_{2,2} & \alpha_{2,3} \\ \alpha_{3,0} & \alpha_{3,1} & \alpha_{3,2} & \alpha_{3,3} \end{pmatrix} \begin{pmatrix} \beta_{0,0} & \beta_{0,1} & \beta_{0,2} & \beta_{0,3} \\ \beta_{1,0} & \beta_{1,1} & \beta_{1,2} & \beta_{1,3} \\ \beta_{2,0} & \beta_{2,1} & \beta_{2,2} & \beta_{2,3} \\ \beta_{3,0} & \beta_{3,1} & \beta_{3,2} & \beta_{3,3} \end{pmatrix} \end{array}\right)$$

(3) C=AB시, 앞 뒤로 다 partition할 때,

$$C = \left(A_0 \mid A_1 \right) \left(\frac{B_0}{B_1} \right) = A_0 B_0 + A_1 B_1$$

$$\left(\begin{array}{cccc} \gamma_{0,0} & \gamma_{0,1} & \gamma_{0,2} & \gamma_{0,3} \\ \gamma_{1,0} & \gamma_{1,1} & \gamma_{1,2} & \gamma_{1,3} \\ \gamma_{2,0} & \gamma_{2,1} & \gamma_{2,2} & \gamma_{2,3} \\ \gamma_{3,0} & \gamma_{3,1} & \gamma_{3,2} & \gamma_{3,3} \end{array}\right) = \left(\begin{array}{cc|cc} \alpha_{0,0} & \alpha_{0,1} & \alpha_{0,2} & \alpha_{0,3} \\ \alpha_{1,0} & \alpha_{1,1} & \alpha_{1,2} & \alpha_{1,3} \\ \alpha_{2,0} & \alpha_{2,1} & \alpha_{2,2} & \alpha_{2,3} \\ \alpha_{3,0} & \alpha_{3,1} & \alpha_{3,2} & \alpha_{3,3} \end{array}\right) \left(\begin{array}{cccc} \beta_{0,0} & \beta_{0,1} & \beta_{0,2} & \beta_{0,3} \\ \beta_{1,0} & \beta_{1,1} & \beta_{1,2} & \beta_{1,3} \\ \hline \beta_{2,0} & \beta_{2,1} & \beta_{2,2} & \beta_{2,3} \\ \beta_{3,0} & \beta_{3,1} & \beta_{3,2} & \beta_{3,3} \end{array}\right) =$$

$$\begin{pmatrix} \alpha_{0,0} & \alpha_{0,1} \\ \alpha_{1,0} & \alpha_{1,1} \\ \alpha_{2,0} & \alpha_{2,1} \\ \alpha_{3,0} & \alpha_{3,1} \end{pmatrix} \begin{pmatrix} \beta_{0,0} & \beta_{0,1} & \beta_{0,2} & \beta_{0,3} \\ \beta_{1,0} & \beta_{1,1} & \beta_{1,2} & \beta_{1,3} \end{pmatrix} + \begin{pmatrix} \alpha_{0,2} & \alpha_{0,3} \\ \alpha_{1,2} & \alpha_{1,3} \\ \alpha_{2,2} & \alpha_{2,3} \\ \alpha_{3,2} & \alpha_{3,3} \end{pmatrix} \begin{pmatrix} \beta_{2,0} & \beta_{2,1} & \beta_{2,2} & \beta_{2,3} \\ \beta_{3,0} & \beta_{3,1} & \beta_{3,2} & \beta_{3,3} \end{pmatrix}$$

(4) C=AB시, A, B를 또 각각 2x2로 쪼개기

$$\left(\begin{array}{c|c} C_{00} & C_{01} \\ \hline C_{10} & C_{11} \end{array}\right) = \left(\begin{array}{c|c} A_{00} & A_{01} \\ \hline A_{10} & A_{11} \end{array}\right)\left(\begin{array}{c|c} B_{00} & B_{01} \\ \hline B_{10} & B_{11} \end{array}\right)$$
$$= \left(\begin{array}{c|c} A_{00}B_{00} + A_{01}B_{10} & A_{00}B_{01} + A_{01}B_{11} \\ \hline A_{10}B_{00} + A_{11}B_{10} & A_{10}B_{01} + A_{11}B_{11} \end{array}\right)$$

$$\left(\begin{array}{cc|cc} \gamma_{0,0} & \gamma_{0,1} & \gamma_{0,2} & \gamma_{0,3} \\ \gamma_{1,0} & \gamma_{1,1} & \gamma_{1,2} & \gamma_{1,3} \\ \hline \gamma_{2,0} & \gamma_{2,1} & \gamma_{2,2} & \gamma_{2,3} \\ \gamma_{3,0} & \gamma_{3,1} & \gamma_{3,2} & \gamma_{3,3} \end{array}\right) = \left(\begin{array}{cc|cc} \alpha_{0,0} & \alpha_{0,1} & \alpha_{0,2} & \alpha_{0,3} \\ \alpha_{1,0} & \alpha_{1,1} & \alpha_{1,2} & \alpha_{1,3} \\ \hline \alpha_{2,0} & \alpha_{2,1} & \alpha_{2,2} & \alpha_{2,3} \\ \alpha_{3,0} & \alpha_{3,1} & \alpha_{3,2} & \alpha_{3,3} \end{array}\right)\left(\begin{array}{cc|cc} \beta_{0,0} & \beta_{0,1} & \beta_{0,2} & \beta_{0,3} \\ \beta_{1,0} & \beta_{1,1} & \beta_{1,2} & \beta_{1,3} \\ \hline \beta_{2,0} & \beta_{2,1} & \beta_{2,2} & \beta_{2,3} \\ \beta_{3,0} & \beta_{3,1} & \beta_{3,2} & \beta_{3,3} \end{array}\right) =$$

$$\left(\begin{array}{c|c} \begin{pmatrix} \alpha_{0,0} & \alpha_{0,1} \\ \alpha_{1,0} & \alpha_{1,1} \end{pmatrix}\begin{pmatrix} \beta_{0,0} & \beta_{0,1} \\ \beta_{1,0} & \beta_{1,1} \end{pmatrix} + \begin{pmatrix} \alpha_{0,2} & \alpha_{0,3} \\ \alpha_{1,2} & \alpha_{1,3} \end{pmatrix}\begin{pmatrix} \beta_{2,0} & \beta_{2,1} \\ \beta_{3,0} & \beta_{3,1} \end{pmatrix} & \begin{pmatrix} \alpha_{0,0} & \alpha_{0,1} \\ \alpha_{1,0} & \alpha_{1,1} \end{pmatrix}\begin{pmatrix} \beta_{0,2} & \beta_{0,3} \\ \beta_{1,2} & \beta_{1,3} \end{pmatrix} + \begin{pmatrix} \alpha_{0,2} & \alpha_{0,3} \\ \alpha_{1,2} & \alpha_{1,3} \end{pmatrix}\begin{pmatrix} \beta_{2,2} & \beta_{2,3} \\ \beta_{3,2} & \beta_{3,3} \end{pmatrix} \\ \hline \begin{pmatrix} \alpha_{2,0} & \alpha_{2,1} \\ \alpha_{3,0} & \alpha_{3,1} \end{pmatrix}\begin{pmatrix} \beta_{0,0} & \beta_{0,1} \\ \beta_{1,0} & \beta_{1,1} \end{pmatrix} + \begin{pmatrix} \alpha_{2,2} & \alpha_{2,3} \\ \alpha_{3,2} & \alpha_{3,3} \end{pmatrix}\begin{pmatrix} \beta_{2,0} & \beta_{2,1} \\ \beta_{3,0} & \beta_{3,1} \end{pmatrix} & \begin{pmatrix} \alpha_{2,0} & \alpha_{2,1} \\ \alpha_{3,0} & \alpha_{3,1} \end{pmatrix}\begin{pmatrix} \beta_{0,2} & \beta_{0,3} \\ \beta_{1,2} & \beta_{1,3} \end{pmatrix} + \begin{pmatrix} \alpha_{2,2} & \alpha_{2,3} \\ \alpha_{3,2} & \alpha_{3,3} \end{pmatrix}\begin{pmatrix} \beta_{2,2} & \beta_{2,3} \\ \beta_{3,2} & \beta_{3,3} \end{pmatrix} \end{array}\right)$$

How to remember

$$\left(\begin{array}{c|c} 1 & -1 \\ \hline 2 & 3 \end{array}\right)\left(\begin{array}{c|c} -2 & 0 \\ \hline 1 & 2 \end{array}\right) = \left(\begin{array}{c|c} (1)\times(-2) + (-1)\times(1) & (1)\times(0) + (-1)\times(2) \\ \hline (2)\times(-2) + (3)\times(1) & (2)\times(0) + (3)\times(2) \end{array}\right)$$

$$\left(\begin{array}{c|c} A_{00} & A_{01} \\ \hline A_{10} & A_{11} \end{array}\right)\left(\begin{array}{c|c} B_{00} & B_{01} \\ \hline B_{10} & B_{11} \end{array}\right) = \left(\begin{array}{c|c} A_{00}B_{00} + A_{01}B_{10} & A_{00}B_{01} + A_{01}B_{11} \\ \hline A_{10}B_{00} + A_{11}B_{10} & A_{10}B_{01} + A_{11}B_{11} \end{array}\right)$$

→ 결국에는 scalar로 이루어진 행렬 곱이랑 형태가 비슷하므로 그것으로 이해하면 문제가 없다.

Let $C \in \mathbb{R}^{m\times n}$, $A \in \mathbb{R}^{m\times k}$, and $B \in \mathbb{R}^{k\times n}$, and $C = AB$.

Partition, conformally,

$$C = \left(\begin{array}{c|c|c|c} C_{0,0} & C_{0,1} & \cdots & C_{0,N-1} \\ \hline C_{1,0} & C_{1,1} & \cdots & C_{1,N-1} \\ \hline \vdots & \vdots & \ddots & \vdots \\ \hline C_{M-1,0} & C_{M-1,1} & \cdots & C_{M-1,N-1} \end{array}\right) A = \left(\begin{array}{c|c|c|c} A_{0,0} & A_{0,1} & \cdots & A_{0,K-1} \\ \hline A_{1,0} & A_{1,1} & \cdots & A_{1,K-1} \\ \hline \vdots & \vdots & \ddots & \vdots \\ \hline A_{M-1,0} & A_{M-1,1} & \cdots & A_{M-1,K-1} \end{array}\right)$$

$$\text{and} \quad B = \left(\begin{array}{c|c|c|c} B_{0,0} & B_{0,1} & \cdots & B_{0,N-1} \\ \hline B_{1,0} & B_{1,1} & \cdots & B_{1,N-1} \\ \hline \vdots & \vdots & \ddots & \vdots \\ \hline B_{K-1,0} & B_{K-1,1} & \cdots & B_{K-1,N-1} \end{array}\right),$$

Then

$$C_{i,j} = \sum_{p=0}^{K-1} A_{i,p}B_{p,j}.$$

5.2. properties

(1) 전치행렬 씌우면 순서가 바뀜

Let $A \in \mathbb{R}^{m\times k}$ and $B \in \mathbb{R}^{k\times n}$. $(AB)^T = B^T A^T$.

Select an option ▾ **Answer:** Always

$$
\begin{aligned}
(AB)^T &= \quad < \text{Partition } A \text{ by rows and } B \text{ by columns} > \\
&\left(\left(\begin{array}{c} \tilde{a}_0^T \\ \hline \tilde{a}_1^T \\ \hline \vdots \\ \hline \tilde{a}_{m-1}^T \end{array} \right) \left(\begin{array}{c|c|c|c} b_0 & b_1 & \cdots & b_{n-1} \end{array} \right) \right)^T \\
&= \quad < \text{Partitioned matrix-matrix multiplication} > \\
&\left(\begin{array}{c|c|c|c} \tilde{a}_0^T b_0 & \tilde{a}_0^T b_1 & \cdots & \tilde{a}_0^T b_{n-1} \\ \hline \tilde{a}_1^T b_0 & \tilde{a}_1^T b_1 & \cdots & \tilde{a}_1^T b_{n-1} \\ \hline \vdots & \vdots & \ddots & \vdots \\ \hline \tilde{a}_{m-1}^T b_0 & \tilde{a}_{m-1}^T b_1 & \cdots & \tilde{a}_{m-1}^T b_{n-1} \end{array} \right)^T \\
&= \quad < \text{Transpose the matrix} > \\
&\left(\begin{array}{c|c|c|c} \tilde{a}_0^T b_0 & \tilde{a}_1^T b_0 & \cdots & \tilde{a}_{m-1}^T b_0 \\ \hline \tilde{a}_0^T b_1 & \tilde{a}_1^T b_1 & \cdots & \tilde{a}_{m-1}^T b_1 \\ \hline \vdots & \vdots & \ddots & \vdots \\ \hline \tilde{a}_0^T b_{n-1} & \tilde{a}_1^T b_{n-1} & \cdots & \tilde{a}_{m-1}^T b_{n-1} \end{array} \right) \\
&= \quad < \text{dot product commutes} > \\
&\left(\begin{array}{c|c|c|c} b_0^T \tilde{a}_0 & b_0^T \tilde{a}_1 & \cdots & b_0^T \tilde{a}_{m-1} \\ \hline b_1^T \tilde{a}_0 & b_1^T \tilde{a}_1 & \cdots & b_1^T \tilde{a}_{m-1} \\ \hline \vdots & \vdots & \ddots & \vdots \\ \hline b_{n-1}^T \tilde{a}_0 & b_{n-1}^T \tilde{a}_1 & \cdots & b_{n-1}^T \tilde{a}_{m-1} \end{array} \right)
\end{aligned}
$$

→ dot product가 commute한다는 것을 이용하는 것이 핵심이다.

$$
\begin{aligned}
&= \quad < \text{Partitioned matrix-matrix multiplication} > \\
&\left(\begin{array}{c} b_0^T \\ \hline b_1^T \\ \hline \vdots \\ \hline b_{n-1}^T \end{array} \right) \left(\begin{array}{c|c|c|c} \tilde{a}_0 & \tilde{a}_1 & \cdots & \tilde{a}_{m-1} \end{array} \right) \\
&= \quad < \text{Partitioned matrix transposition} > \\
&\left(\begin{array}{c|c|c|c} b_0 & b_1 & \cdots & b_{n-1} \end{array} \right)^T \left(\begin{array}{c} \tilde{a}_0^T \\ \hline \tilde{a}_1^T \\ \hline \vdots \\ \hline \tilde{a}_{m-1}^T \end{array} \right)^T = B^T A^T.
\end{aligned}
$$

→ 응용

Let $A \in \mathbb{R}^{m \times n}$.

AA^T is symmetric.

Select an option ▾ **Answer:** Always

Explanation

Answer: Always

Proof 1: $(AA^T)^T = (A^T)^T A^T = AA^T$.

(2) Diagonal을 사용하면 A에 각 행/열벡터에 원하는 값을 곱할 수 있다.

Let $A \in \mathbb{R}^{m \times n}$ and let D denote the diagonal matrix with diagonal elements $\delta_0, \delta_1, \cdots, \delta_{n-1}$. Partition A by columns :

$$A = \left(\begin{array}{c|c|c|c} a_0 & a_1 & \cdots & a_{n-1} \end{array} \right).$$

$$AD = \left(\begin{array}{c|c|c|c} \delta_0 a_0 & \delta_1 a_1 & \cdots & \delta_{n-1} a_{n-1} \end{array} \right).$$

Let $A \in \mathbb{R}^{m \times n}$ and let D denote the diagonal matrix with diagonal elements $\delta_0, \delta_1, \cdots, \delta_{m-1}$. Partition A by rows :

$$A = \left(\begin{array}{c} \tilde{a}_0^T \\ \hline \tilde{a}_1^T \\ \hline \vdots \\ \hline \tilde{a}_{m-1}^T \end{array} \right).$$

$$DA = \left(\begin{array}{c} \delta_0 \tilde{a}_0^T \\ \hline \delta_1 \tilde{a}_1^T \\ \hline \vdots \\ \hline \delta_{m-1} \tilde{a}_{m-1}^T \end{array} \right).$$

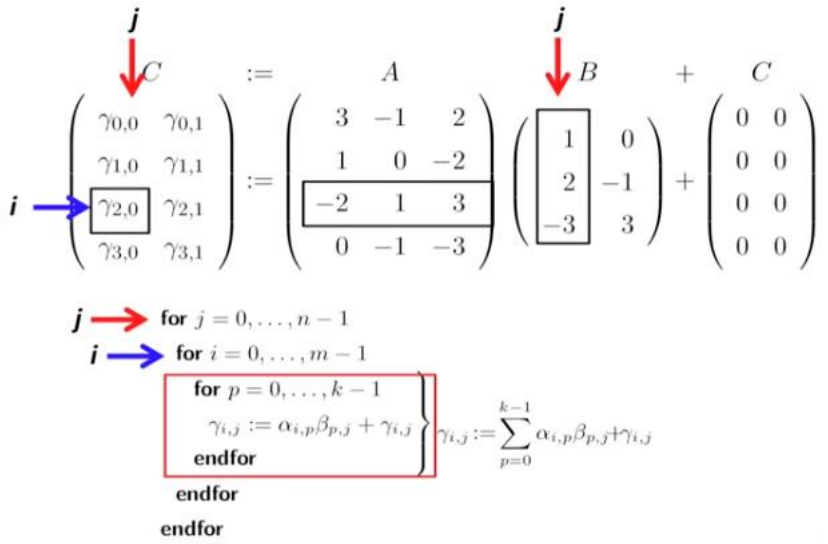
5.3. 알고리즘

→ matrix-matrix multiplication은 triple-nested loop이다.

Algorithms for computing matrix-matrix multiplication

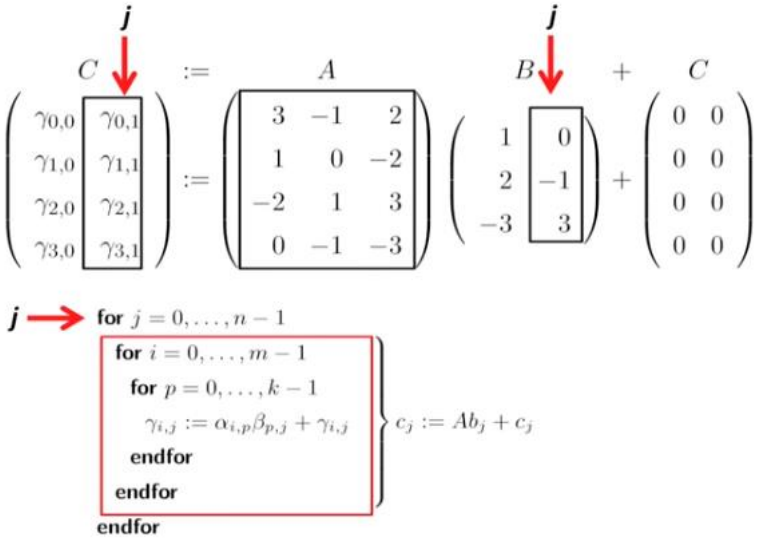
for $i = 0, \ldots, m-1$
  for $j = 0, \ldots, n-1$
    for $p = 0, \ldots, k-1$
      $\gamma_{i,j} := \alpha_{i,p}\beta_{p,j} + \gamma_{i,j}$
    endfor
  endfor
endfor

(1) matrix-matrix multiplication by columns

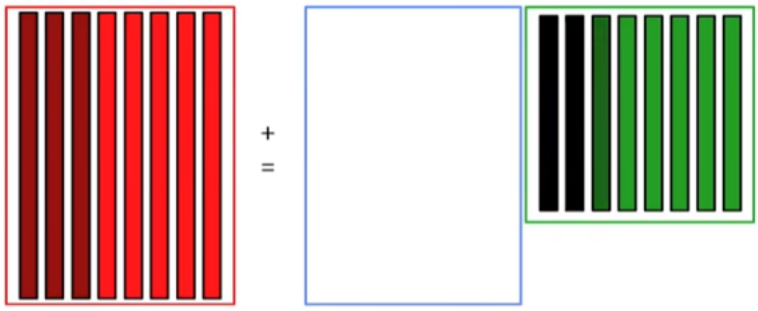


→ dot product를 하면 triple loop을 double loop으로 줄일 수 있다.

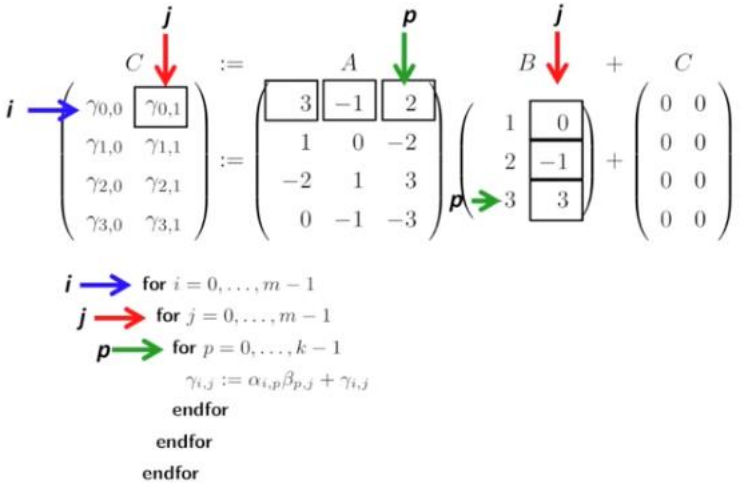
- 행렬 A 단위로 하면 j loop만 하면 된다.



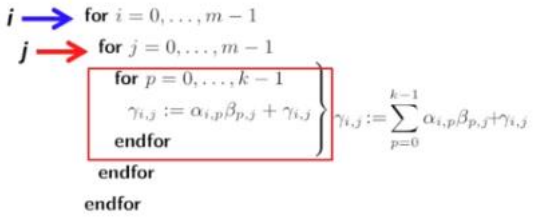
Summary



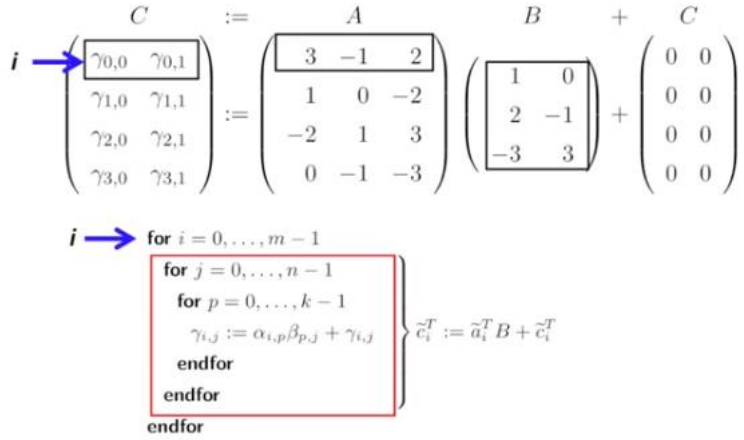
(2) Matrix-matrix multiplication by rows



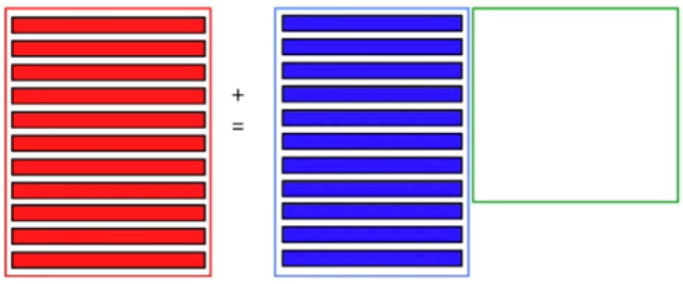
→ dot routine으로 하면 double loop이 된다.



- Matrix로 하면 single loop이 된다.



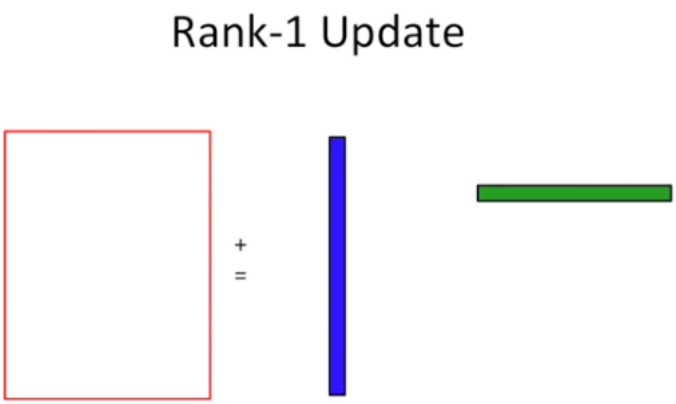
Summary



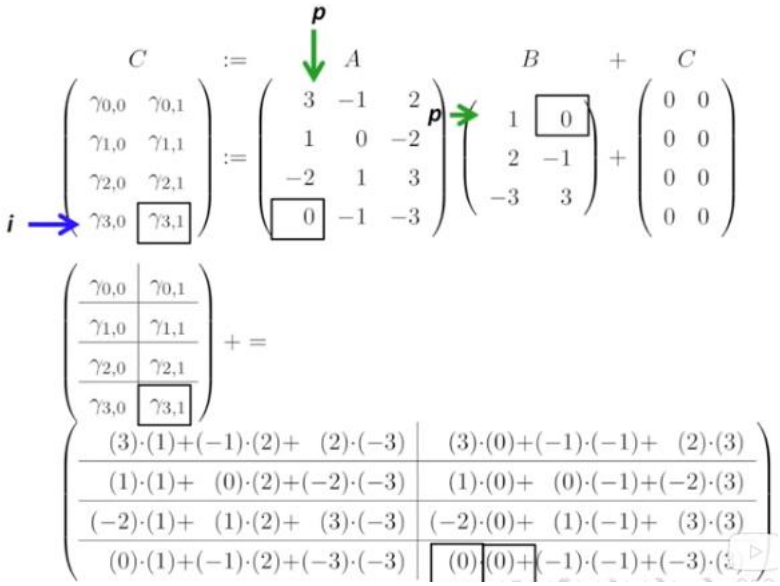
(3) Matrix-matrix multiplication with Rank-1 updates

- 아까는 I, j loop을 밖으로 빼냈는데, P loop이 밖에 있으면 어떻게 될까?

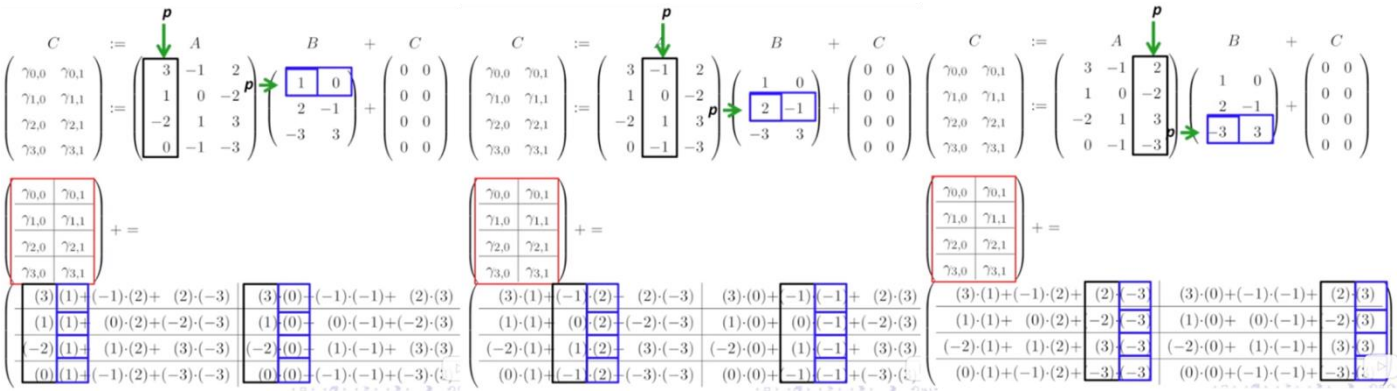
- rank 1 update 정의 : outer product(column vector x row vector)가 matrix를 update하게 되면!

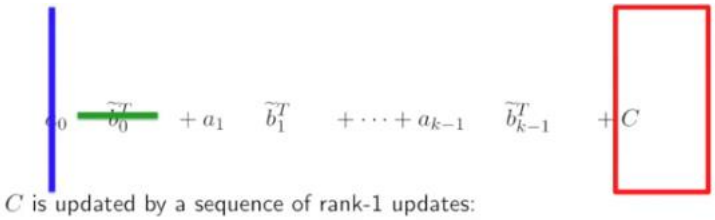


→ P가 고정되면 다른 I, j가 막 움직인다. P는 A1열-B1행, A2열-B2행, A3열-B3행 방식이 있다. 각각 8번을 진행한다.



- single loop으로 처리하게 되면 outer product를 만들어서 행렬 3개를 더하는 꼴이 된다.



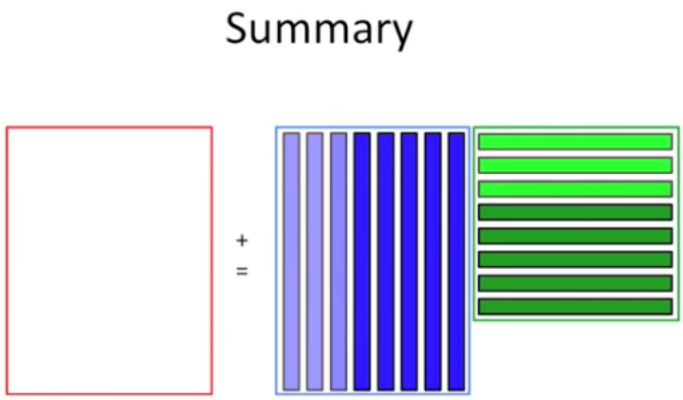


C is updated by a sequence of rank-1 updates:

for $p = 0, \ldots, k-1$

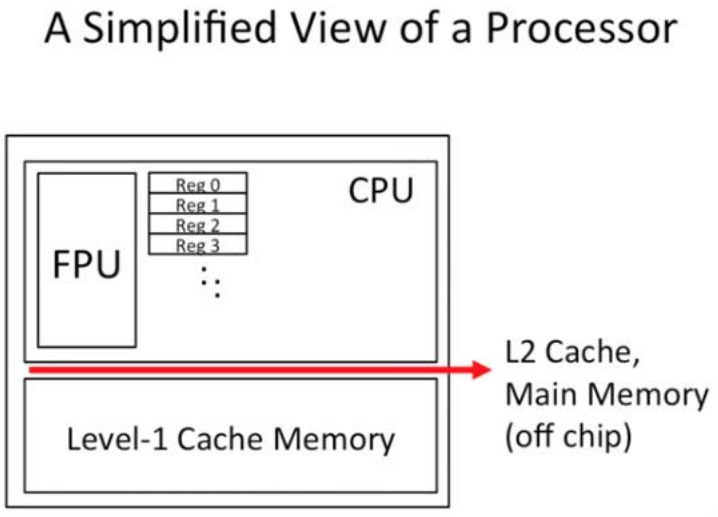
$$C := a_p \tilde{b}_p^T + C$$

endfor

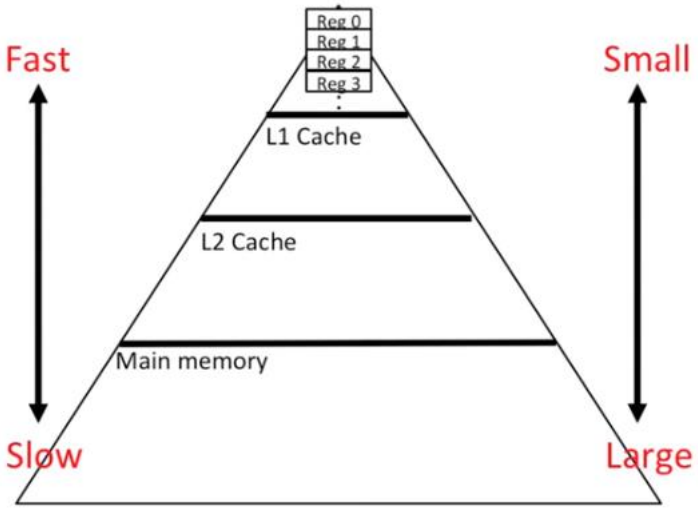


5.4. Slicing and dicing for performance

- 컴퓨터 구조

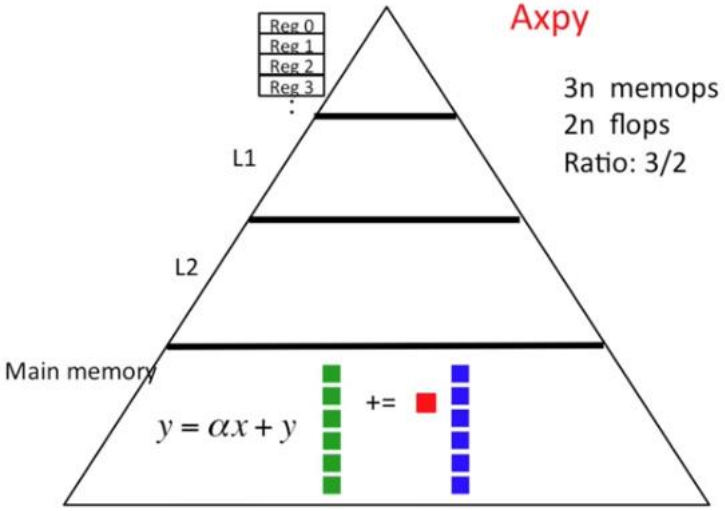


→ Level 1(on chip) 다 차면 level 2 cache로, 다 차면 Main memory로

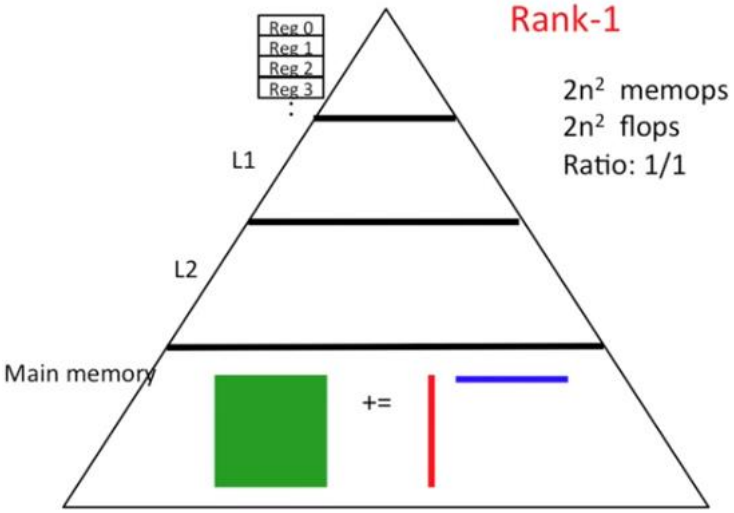


→ register가 가장 빠르다. Main memory에서 cache로 정보를 가져오는 것이 real bottleneck이다.

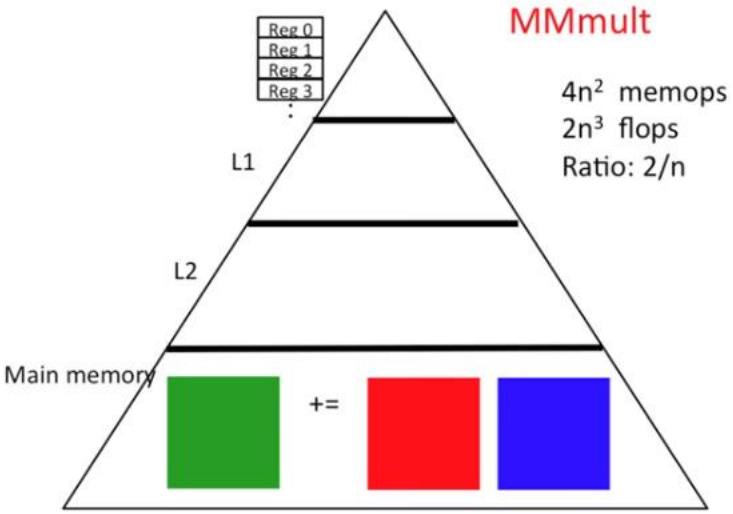
- 예



- 예2 : Rank 1 update : 초록색 행렬이 더해지면 다시 memory로 update된 것을 저장해야 하기 때문에 $2n^2$이다.

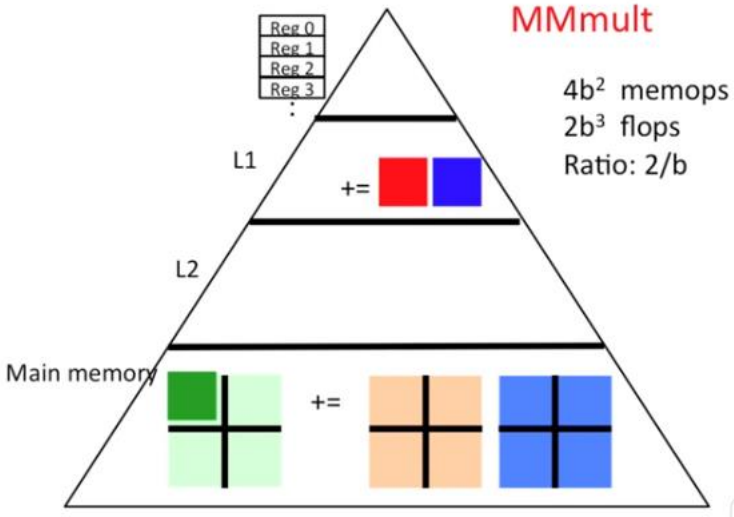


- 예3 : matrix-vector multiplication



→ matrix가 3개를 읽어야 하고 그린을 다시 update해서 저장해야 하기 때문에 $4n^2$이다.

- partition하는 것도 가능하다. 그러면 L1에 저장이 가능해진다. 작기 때문에. Matrix 통째로는 cache에 안들어간다.



6. Gaussian Elimination

6.2.1. Reducing a System of Linear Equations to an Upper Triangular System

6.2.2. Appended Matrices

6.2.3. Gauss Transforms

6.2.4. Computing Separately with the Matrix and Right-Hand Side (Forward Substitution)

6.2.5. Towards an Algorithm

Solving Ax = b via LU Factorization

6.3.1. LU factorization (Gaussian elimination)

6.3.2. Solving Lz = b (Forward substitution)

6.3.3. Solving Ux = b (Back substitution)

6.3.4. Putting it all together to solve Ax = b

6.3.5. Cost

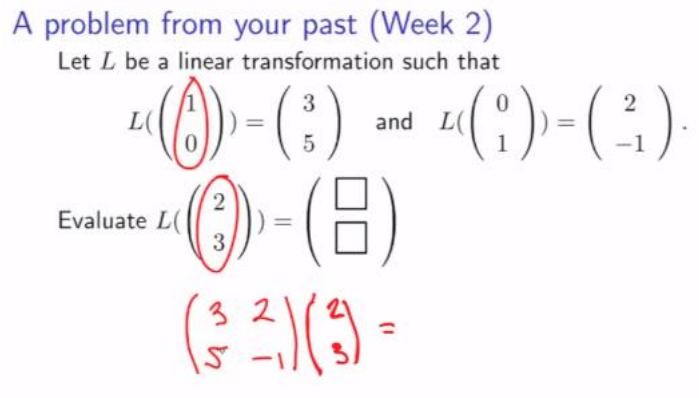
Enrichment

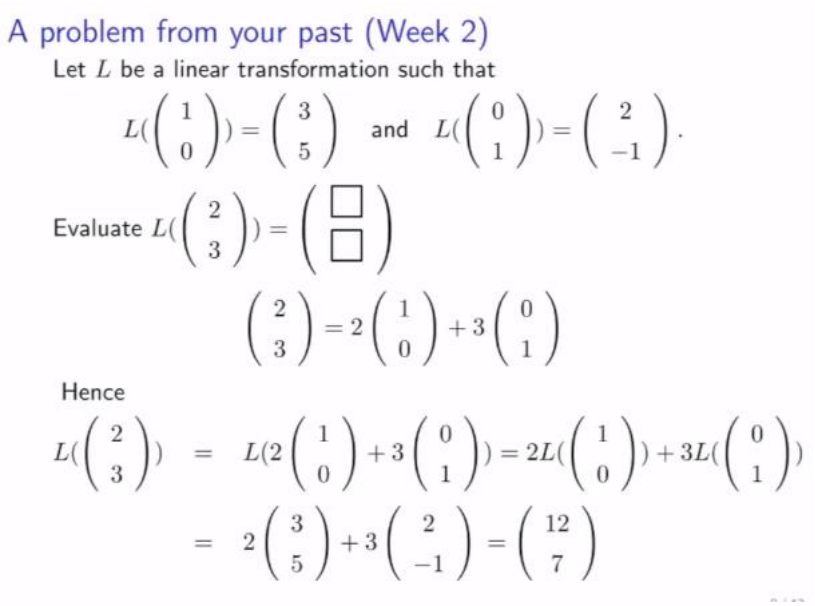
6.4.1. Blocked LU Factorization

6. Gaussian Elimination

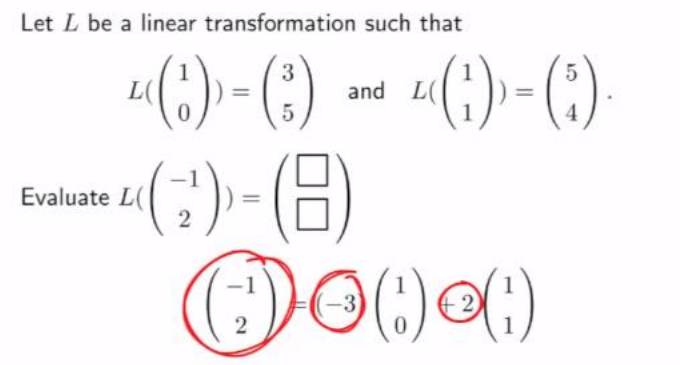
- 지금까지 vector와 행렬에 대한 친숙도를 높여왔다. Slicing도 잘 할 줄 알게 됐다.
- 지금부터는 선형대수학에 관련된 문제 Ax=b를 본격적으로 푼다.
- 한 문제를 푸는 방법이 여러가지가 있다.

6.1. 일반적인 문제풀이법 : 연립방정식 풀기





- 0과 1이 없는 것은 더 어렵다.



→ 더 일반화해서 푸는 방법은

$$\begin{pmatrix} -1 \\ 2 \end{pmatrix} = \chi_0 \begin{pmatrix} 1 \\ 0 \end{pmatrix} + \chi_1 \begin{pmatrix} 1 \\ 1 \end{pmatrix}$$

$$\begin{pmatrix} -1 \\ 2 \end{pmatrix} = \chi_0 \begin{pmatrix} 1 \\ 0 \end{pmatrix} + \chi_1 \begin{pmatrix} 1 \\ 1 \end{pmatrix}$$

$$\chi_0 \begin{pmatrix} 1 \\ 0 \end{pmatrix} + \chi_1 \begin{pmatrix} 1 \\ 1 \end{pmatrix} = \begin{pmatrix} -1 \\ 2 \end{pmatrix}$$

$$(1) \times \chi_0 + (1) \times \chi_1 = -1$$
$$(0) \times \chi_0 + (1) \times \chi_1 = 2$$

$$\chi_0 + \cancel{\chi_1}\ 2 = -1 \qquad \chi_0 = -3$$
$$\chi_1 = 2$$

- 더 복잡해질 수도 있다.... 연립방정식으로 계속 풀 것인가?

$$\begin{pmatrix} -1 \\ 0 \end{pmatrix} = \chi_0 \begin{pmatrix} 1 \\ 2 \end{pmatrix} + \chi_1 \begin{pmatrix} 1 \\ 3 \end{pmatrix}$$

$$\chi_0 \begin{pmatrix} 1 \\ 2 \end{pmatrix} + \chi_1 \begin{pmatrix} 1 \\ 3 \end{pmatrix} = \begin{pmatrix} -1 \\ 0 \end{pmatrix}$$

$$(1) \times \chi_0 + (1) \times \chi_1 = -1$$
$$(2) \times \chi_0 + (3) \times \chi_1 = 0$$

$$\chi_0 + \chi_1 = -1$$
$$2\chi_0 + 3\chi_1 = 0 \qquad \chi_1 = -2\chi_0/3$$
$$\chi_0 - 2\chi_0/3 = -1$$

→ 따라서, Gaussian elimination을 하면 된다.

$$2\times \longrightarrow \chi_0 + \chi_1 = -1$$
$$\longrightarrow 2\chi_0 + 3\chi_1 = 0$$

$$\chi_0 + \chi_1 = -1$$
$$\chi_1 = 2$$

$$\chi_0 + 2 = -1 \Rightarrow \chi_0 = -3$$

→ 위의 식을 2배 해서 아래식에서 이를 빼준다.

- 여러가지 표현 방법 : Appended matrices

$$\begin{aligned} \chi_0 + \chi_1 &= -1 \\ 2\chi_0 + 3\chi_1 &= 0 \end{aligned}$$

$$\begin{pmatrix} 1 & 1 \\ 2 & 3 \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} = \begin{pmatrix} -1 \\ 0 \end{pmatrix}$$

$$\left(\begin{array}{cc|c} 1 & 1 & -1 \\ 2 & 3 & 0 \end{array}\right)$$

→ 왜 appended matrix로 바꾸는가. Variable을 저장할 필요가 없고 단순화하여 upper triangular system을 빠르게 만들 수 있기 때문이다.

6.2. 좀 더 일반화한 풀이법 : Reducing a System of Linear Equations to an Upper Triangular System

➔ Upper triangle system으로 가기 위한 방법이 바로 "Gauss transforms"

Upper triangular system이 푸는게 더 용이하다.

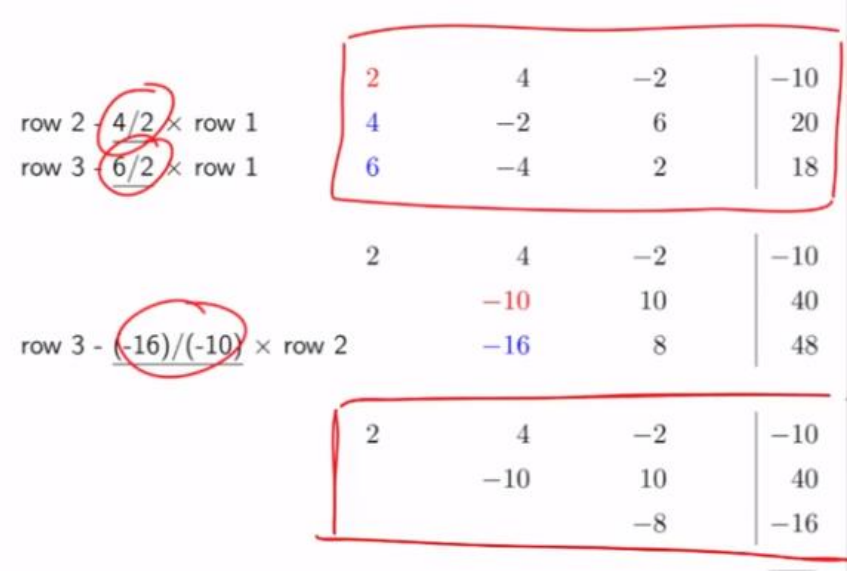
Ax=B를 푸는 방법 중 **Gaussian elimination**의 3단계

1. **gauss transform** → linear equation을 upper triangle system으로 만든다. 양변에 gauss transform 행렬을 곱하면 가능하다.
2. **forward substitution** → b_1, b_2, b_3, ..., b_n 의 right hand side를 구하는 것. 왼쪽 변을 gauss transform 행렬을 곱했으니 오른쪽 변도 곱해야 한다.
3. **backward substitution** → x_1, x_2, x_3 의 해를 구하는 과정 (왜냐하면 gauss transformation을 해서 upper unit trangular을 만들었기 때문에 가능하다. B로 알 뿐만 아니다.

6.2.1. 문제 정의 : Ax=B를 appended matrices로 나타내기

$$\begin{aligned} 2\chi_0 + 4\chi_1 - 2\chi_2 &= -10 \\ 4\chi_0 - 2\chi_1 + 6\chi_2 &= 20 \\ 6\chi_0 - 4\chi_1 + 2\chi_2 &= 18 \end{aligned}$$

- 소거해서 빼준다.



→ Reducing the linear equations : 두번째 세번째 줄을 16/10해서 빼면 세번째줄의 항이 1개가 된다. 마지막 3개의 식이 upper triangular system이라고 볼 수 있다.

→ 위의 식을 아래와 같이 바꿀 수 있다.

6.2.3 Gauss transforms : 대각선은 1인데 그 왼쪽에 수가 있는 것을 곱하는 것이 gauss transform이다.

Gauss transforms

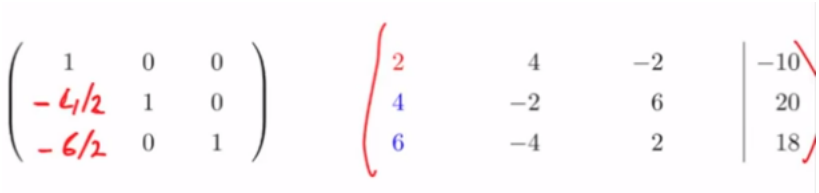
A Gauss transform is a matrix of the form

$$L_j = \begin{pmatrix} I_j & 0 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & 0 & \cdots & 0 \\ 0 & -\lambda_{j+1,j} & 1 & 0 & \cdots & 0 \\ 0 & -\lambda_{j+2,j} & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & -\lambda_{n-1,j} & 0 & 0 & \cdots & 1 \end{pmatrix}.$$

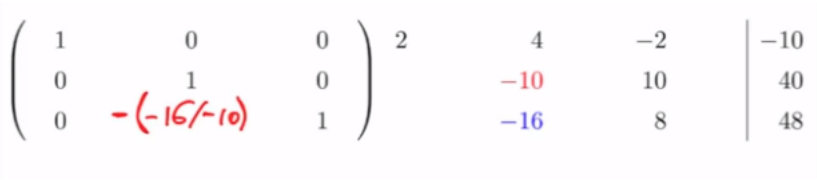
When applied to a matrix (or a vector, which we think of as a special case of a matrix), it subtracts $\lambda_{i,j}$ times the jth row from the ith row, for $i = j+1, \ldots, n-1$. Gauss transforms can be used to express the operations that are inherently performed as part of Gaussian elimination with an appended system of equations.

The action of a Gauss transform on a matrix, $A := L_j A$ can be described as follows:

$$\begin{pmatrix} I_j & 0 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & 0 & \cdots & 0 \\ 0 & -\lambda_{j+1,j} & 1 & 0 & \cdots & 0 \\ 0 & -\lambda_{j+2,j} & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & -\lambda_{n-1,j} & 0 & 0 & \cdots & 1 \end{pmatrix} \begin{pmatrix} A_0 \\ \widetilde{a}_j^T \\ \widetilde{a}_{j+1}^T \\ \vdots \\ \widetilde{a}_{n-1}^T \end{pmatrix} = \begin{pmatrix} A_0 \\ \widetilde{a}_j^T \\ \widetilde{a}_{j+1}^T - \lambda_{j+1,j}\widetilde{a}_j^T \\ \vdots \\ \widetilde{a}_{n-1}^T - \lambda_{n-1,j}\widetilde{a}_j^T \end{pmatrix}.$$



→ 오 개신기하다. Gauss transformation을 하기 위해서 gauss transform을 곱한다는 것은 "첫번째 행을 -4/2해서 2번째 행에 더하라". "첫번째 행을 -6/2해서 세번째 행에 더하라."가 된다.

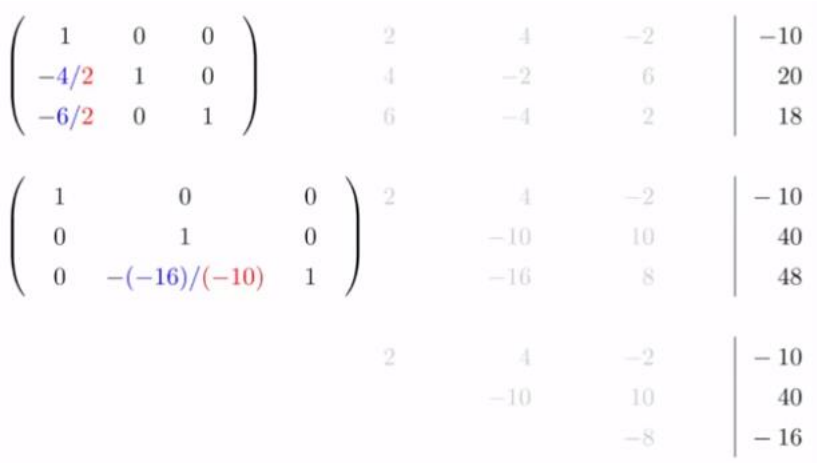


→ 두번째 행을 -16/10배 해서 세번째 행과 더하라.

➔ 결론적으로는

- 다음에 back substitution을 해서 psi2부터 대입해서 해를 풀어나가면 된다.

6.2.4. Computing Separately with the Matrix and Right-Hand Side (Forward Substitution)



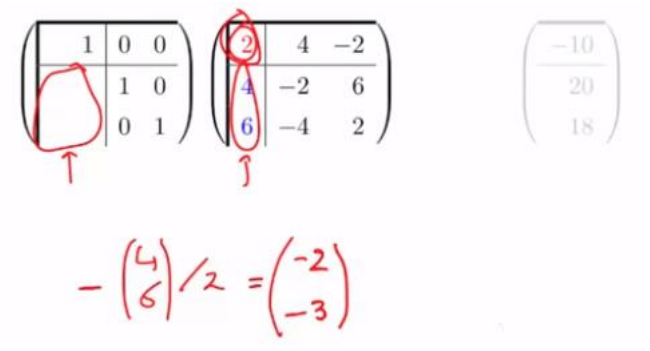
→ 왼쪽을 행렬을 곱했으니, 오른쪽도 행렬을 곱해줘야 할 것이다. 오른쪽이 바로 right rand side를 의미하는 것이다. (이 과정을 바로 forward substitution을 의미한다.)

6.2.5. 해를 구하기 : Backward substitution

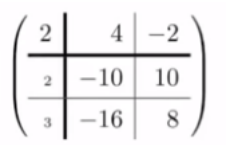
x_n부터 구하고 대입해서 x_{n-1}구하고 계속 해서 x_1까지 구한다.

6.2.6. Algorithm을 만들어보자 : Computing mechanism.

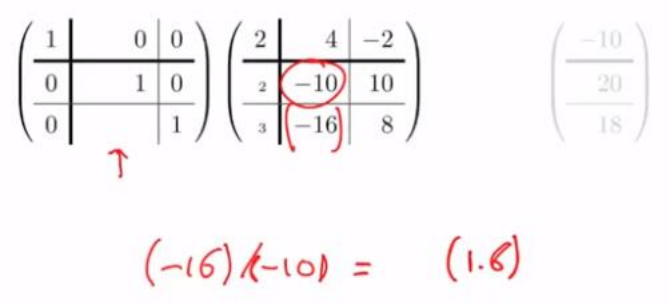
- 1단계 : 왼쪽 아래 가장자리에 넣을 vector를 구한다.



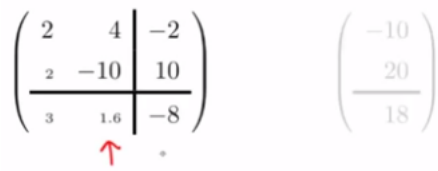
- 2단계 : 2, 3 vector를 안 쓰는 자리에다가 저장을 한다.



- 3단계 : 다시 gause transform을 한다. (아래 가운데 자리 처리)

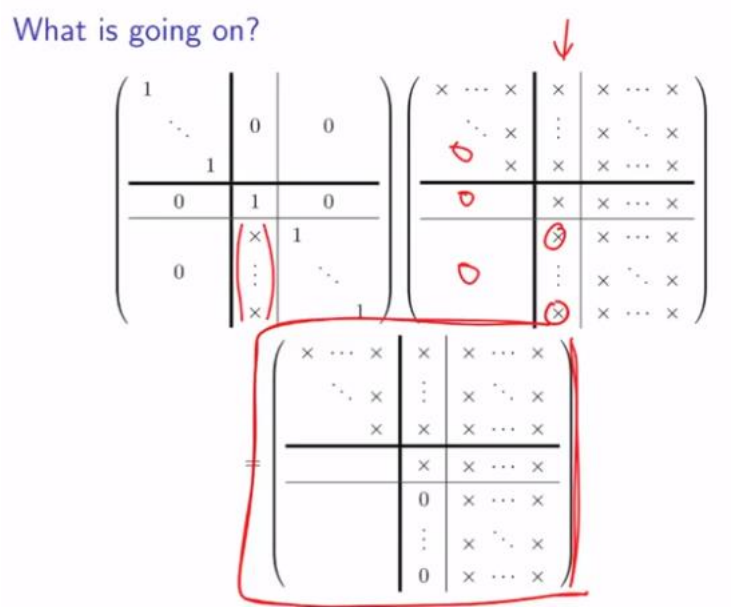


- 4단계 : 안 쓰는 자리에다가 1.6을 저장을 한다.(override the original matrix)



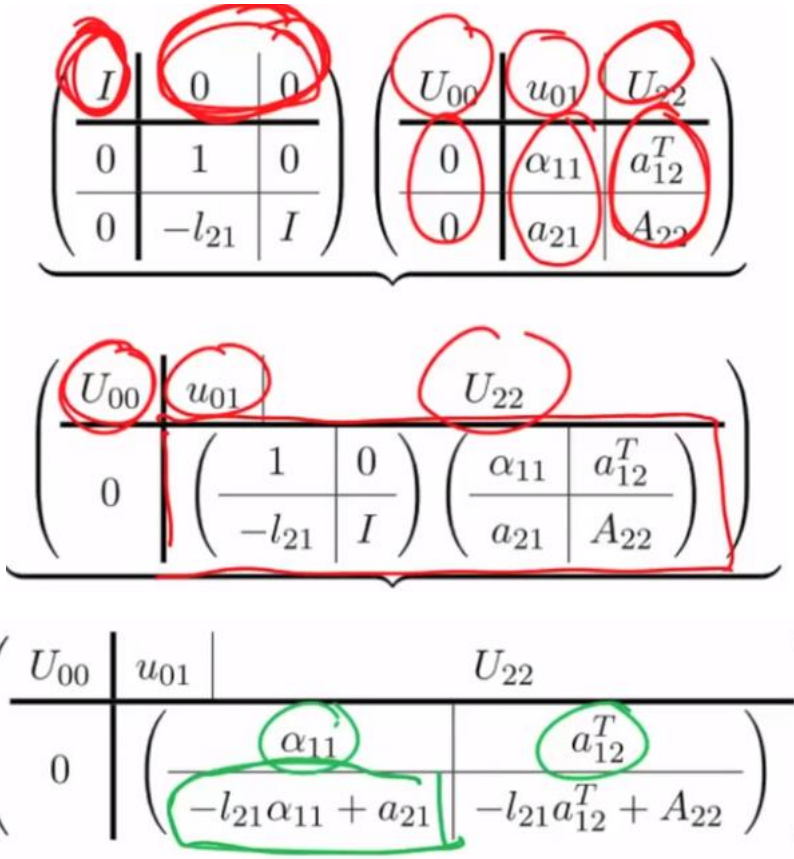
- 4단계 : 저장된 2,3,1.6 정보가 있으니 forward substitution을 할 수 있게 된다.

- 일반화



→ 어떤 가운데 아래의 vector를 구해야지 그 행렬을 곱하게 되면 그 아래가 모두 0이 될 수 있을까?

- Partition multiplication : slicing이랑 같은 방법이다.



→ 여기서 (3, 2)의 위치에 있는 친구가 0이 되게 하면 된다. 알파들은 알고 있으므로 I은 바로 구할 수가 있는 것이다.

<정리>

(1) 1단계 : gauss transform을 slicing(partition multiplication)으로 쭉 한다.

Algorithm: $A := \text{GAUSSIAN\_ELIMINATION}\ (A)$

Partition $A \rightarrow \left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right)$

where A_{TL} is 0×0

while $m(A_{TL}) < m(A)$ **do**

Repartition

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right)$$

$a_{21} := a_{21}/\alpha_{11} \quad (= l_{21})$

$A_{22} := A_{22} - a_{21}a_{12}^T \quad (= A_{22} - l_{21}a_{12}^T)$

Continue with

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right)$$

endwhile

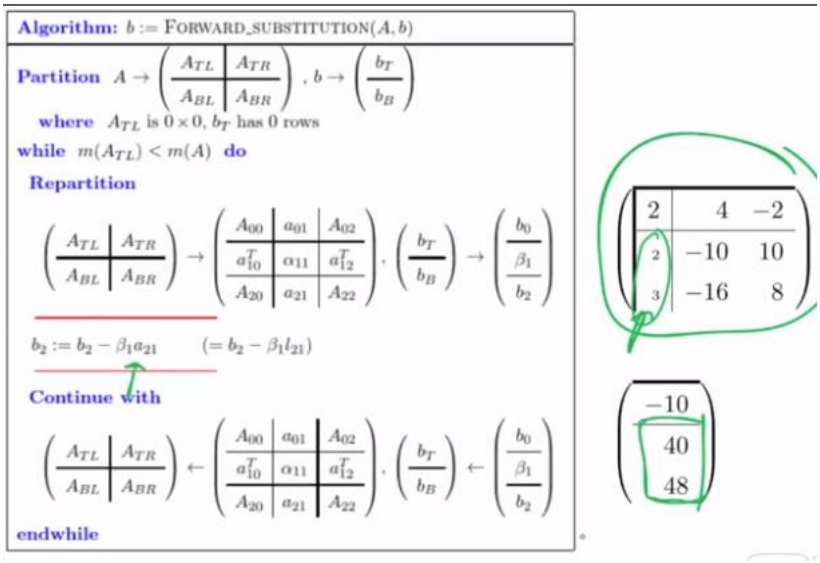
$$\left(\begin{array}{c|cc} 2 & 4 & -2 \\ \hline 2 & -10 & 10 \\ 3 & -16 & 8 \end{array}\right)$$

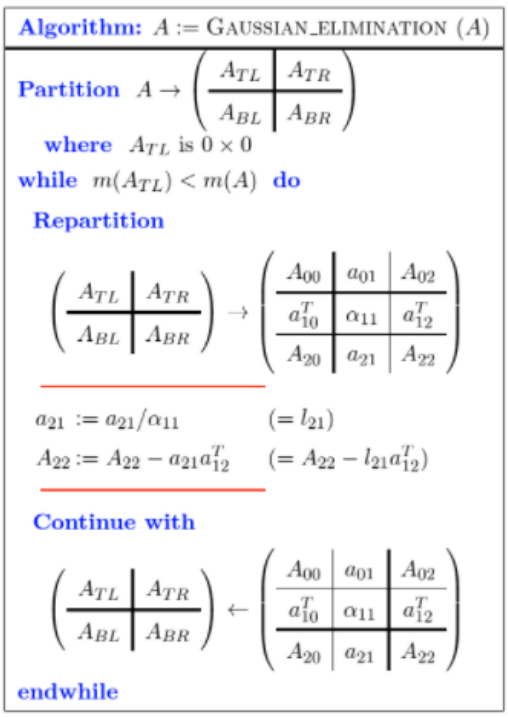
(2) forward substitution로 gauss transform된 B도 구한다.

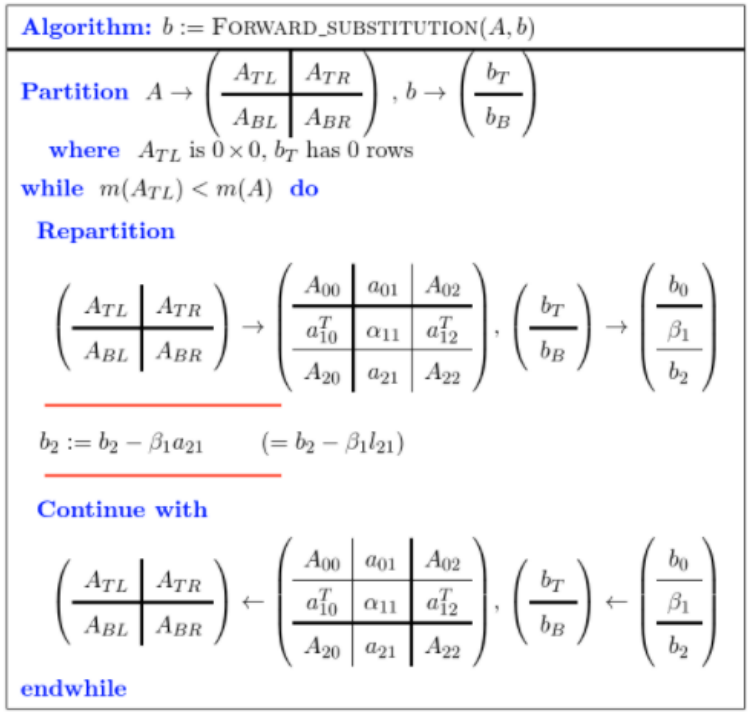
Forward substitution

$$\underbrace{\left(\begin{array}{c|c|c} I & 0 & 0 \\ \hline 0 & 1 & 0 \\ \hline 0 & -l_{21} & I \end{array}\right)\left(\begin{array}{c} y_0 \\ \hline \beta_1 \\ \hline b_2 \end{array}\right)}_{\left(\begin{array}{c} y_0 \\ \hline \left(\begin{array}{c|c} 1 & 0 \\ \hline -l_{21} & I \end{array}\right)\left(\begin{array}{c} \beta_1 \\ \hline b_2 \end{array}\right) \end{array}\right)} = \left(\begin{array}{c} y_0 \\ \hline \beta_1 \\ \hline b_2^{\text{new}} \end{array}\right)$$

$$\left(\begin{array}{c} y_0 \\ \hline \left(\begin{array}{c} \beta_1 \\ \hline -l_{21}\beta_1 + b_2 \end{array}\right) \end{array}\right)$$







→ 알고리즘은 Gaussian elimination에서 gauss transform 과정이랑 forward substitution이랑 유사하다.

(3) 마지막에 X(해)를 구하기 위해 backward substitution을 통해서 Xn부터 대입해서 X1까지 구한다.

6.3. LU factorization

6.3.1. 의미

- Gaussian elimination이랑 같은 과정이다. 그러나 A=LU로 쪼개면 알고리즘이 더 간단해지는 장점이 있다.

- 지금까지의 slicing과 dicing은 이것을 하기 위함이었다!!

- Let $A \in \mathbb{R}^{n \times n}$.
- Under conditions to be discussed later, there exist
  - Unit lower triangular matrix $L \in \mathbb{R}^{n \times n}$; and
  - Upper triangular matrix $U \in \mathbb{R}^{n \times n}$

such that

$$A = LU$$

This is known as the LU factorization of a matrix A: Given A, we compute L and U.

6.3.2. LU factorization으로 문제를 푸는 단계

Ax=B를 푸는 방법 중 **LU factorization**의 3단계
L(Ux) = b ← Ux = z
1. LU factorization해서 나눈다.
2. Lz=b [lazybody ㅋㅋ]를 푼다. (z = Ux) → z 구하기 / 이 과정에서 b를 구하기 때문에 Forward substitution이랑 같은 과정이라고 볼 수 있다.
3. Ux=z를 푼다. → x 구하기 .... 이 과정은 backward substitution이랑 같은 과정이라고 볼 수 있다.

Ax=B를 푸는 방법 중 **Gaussian elimination**의 3단계

1. **Gauss transform** → linear equation을 upper triangle system으로 만든다. 양변에 gauss transform 행렬을 곱하면 가능하다.
2. **Forward substitution** → b_1, b_2, b_3, ..., b_n 의 right hand side를 구하는 것. 왼쪽 변을 gauss transform 행렬을 곱했으니 오른쪽 변도 곱해야 한다.
3. **Backward substitution** → x_1, x_2, x_3 의 해를 구하는 과정 (왜냐하면 gauss transformation을 해서 upper unit trangular을 만들었기 때문에 가능하다. B로 알 뿐만 아니다.

Where is this going?

Want to solve:	$Ax = b.$
We can now find triangular L and U so that	$A = LU.$
Substitute:	$(LU)x = b.$
Matrix multiplication is associative:	$L(Ux) = b.$
We don't know x but we can create a dummy vector $y = Ux.$	$L\ y\ = b.$ Ux
Solve Solving a (lower) triangular system is easy!	$Ly = b$ for y
Solve Solving an (upper) triangular system is easy!	$Ux = y$ for x

→ LU로 쪼개고, y로 치환해서 y구하고 다음에 x를 구하면 된다. Upper triangle, lower trangle은 backward substitution을 사용해서 해를 대입해나가면서 풀 수 있기 때문에 쉽게 풀린다.

(1) LU로 쪼개보기

- 한번 L과 U를 곱해보자.

Partition

$$A \rightarrow \left(\begin{array}{c|c} \alpha_{11} & a_{12}^T \\ \hline a_{21} & A_{22} \end{array}\right), L \rightarrow \left(\begin{array}{c|c} 1 & 0 \\ \hline l_{21} & L_{22} \end{array}\right), U \rightarrow \left(\begin{array}{c|c} \upsilon_{11} & u_{12}^T \\ \hline 0 & U_{22} \end{array}\right)$$

Consider

$$A = LU$$

Substitute:

$$\left(\begin{array}{c|c} \alpha_{11} & a_{12}^T \\ \hline a_{21} & A_{22} \end{array}\right) = \left(\begin{array}{c|c} 1 & 0 \\ \hline l_{21} & L_{22} \end{array}\right)\left(\begin{array}{c|c} \upsilon_{11} & u_{12}^T \\ \hline 0 & U_{22} \end{array}\right)$$

Partition

$$A \rightarrow \left(\begin{array}{c|c} \alpha_{11} & a_{12}^T \\ \hline a_{21} & A_{22} \end{array}\right), L \rightarrow \left(\begin{array}{c|c} 1 & 0 \\ \hline l_{21} & L_{22} \end{array}\right), U \rightarrow \left(\begin{array}{c|c} \upsilon_{11} & u_{12}^T \\ \hline 0 & U_{22} \end{array}\right).$$

Consider

$$A = LU$$

Substitute:

$$\left(\begin{array}{c|c} \alpha_{11} & a_{12}^T \\ \hline a_{21} & A_{22} \end{array}\right) = \underbrace{\left(\begin{array}{c|c} 1 & 0 \\ \hline l_{21} & L_{22} \end{array}\right)\left(\begin{array}{c|c} \upsilon_{11} & u_{12}^T \\ \hline 0 & U_{22} \end{array}\right)}$$

$$\left(\begin{array}{c|c} 1 \times \upsilon_{11} + 0 \times 0 & 1 \times u_{12}^T + 0 \times U_{22} \\ \hline l_{21} \times \upsilon_{11} + L_{22} \times 0 & l_{21} \times u_{12}^T + L_{22}U_{22} \end{array}\right)$$

$$\left(\begin{array}{c|c} \alpha_{11} = \upsilon_{11} & a_{12}^T = u_{12}^T \\ \hline a_{21} = l_{21}\upsilon_{11} & A_{22} = l_{21}u_{12}^T + L_{22}U_{22} \end{array}\right)$$

→ 각각에 대해서 살펴보자. 우리가 아는 것은 A행렬이다. 따라서 a_{11}을 알기 때문에 (0,0)위치는 안다. a_{12}를 알기 때문에 (0,1)위치도 해결이 된다. (1, 0)위치의 l_{21}도 우리가 아는 것을 통해서 충분히 계산할 수 있다. a_{21}과 upsilon11으로 l_{21}을 구할 수 있다.

Partition

$$A \rightarrow \left(\begin{array}{c|c} \alpha_{11} & a_{12}^T \\ \hline a_{21} & A_{22} \end{array}\right), L \rightarrow \left(\begin{array}{c|c} 1 & 0 \\ \hline l_{21} & L_{22} \end{array}\right), U \rightarrow \left(\begin{array}{c|c} \upsilon_{11} & u_{12}^T \\ \hline 0 & U_{22} \end{array}\right).$$

$$A = LU$$

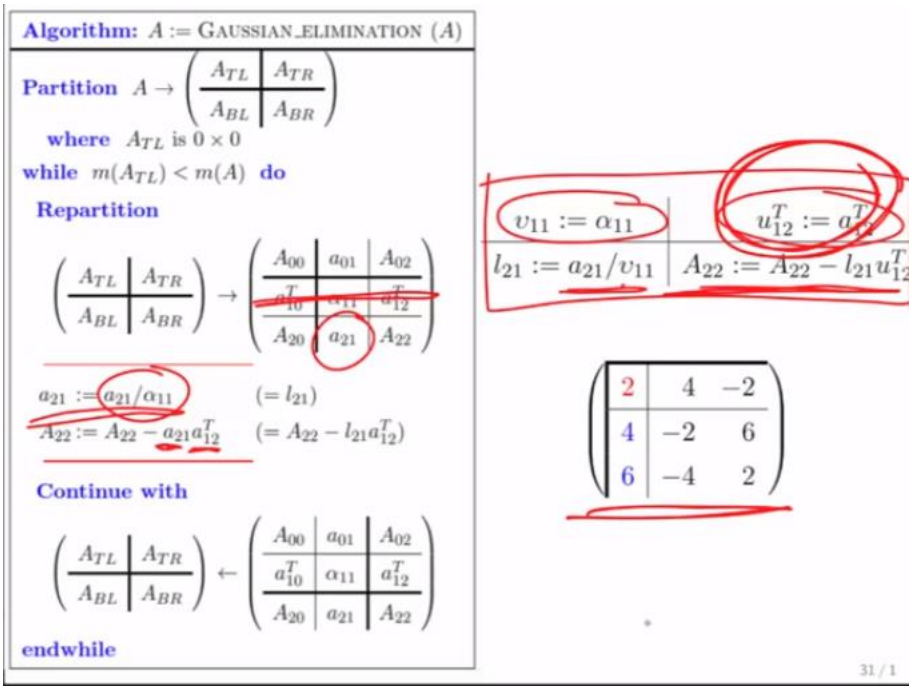
$$\left(\begin{array}{c|c} \alpha_{11} = \upsilon_{11} & a_{12}^T = u_{12}^T \\ \hline a_{21} = l_{21}\upsilon_{11} & A_{22} = l_{21}u_{12}^T + L_{22}U_{22} \end{array}\right)$$

$$\left(\begin{array}{c|c} \upsilon_{11} := \alpha_{11} & u_{12}^T := a_{12}^T \\ \hline l_{21} := a_{21}/\upsilon_{11} & A_{22} - l_{21}u_{12}^T \rightarrow L_{22}U_{22} \end{array}\right)$$

→ a22가 문제인다. 이것을 알려면 $L_{22}U_{22}$를 알아야 한다. 그것을 알기 위해서는 A_{22}행렬에서 $-l_{21}u_{12}^T$를 빼면된다. 이것은 **rank-1 update**라고 한다. 그 이유는 행렬에서 outer product을 빼야 하기 때문이다.(column vector times row vector is an outer product.)

* rank 1 update 정의 : outer product(column vector x row vector)가 matrix를 update하게 되면!

- 알고리즘



→ 오 생각해보니까, LU factorization은 아까 위에서 했던 gaussian elimination방식이랑 완전히 같구나!!

(혁동님) LU로 쪼갠다.

- Gaussian transformation과의 관계는? Gaussian을 써서 U를 구할 수 있다. 그러면 U를 먼저 구하고 L을 구한다 (L은 대각선이 1이므로!!!)

→ 어쨌든 간에 LU transformation을 하면 gauss transform을 할 필요가 없다. 사실.

(2) Ly=b를 푸는 방법(unit lower triangle)

Partition

$$L \rightarrow \left(\begin{array}{c|c} 1 & 0 \\ \hline l_{21} & L_{22} \end{array}\right), z \rightarrow \left(\begin{array}{c} \zeta_1 \\ \hline z_2 \end{array}\right), b \rightarrow \left(\begin{array}{c} \beta_1 \\ \hline b_2 \end{array}\right)$$

Consider

$$Lz = b$$

Substitute:

$$\underbrace{\left(\begin{array}{c|c} 1 & 0 \\ \hline l_{21} & L_{22} \end{array}\right)\left(\begin{array}{c} \zeta_1 \\ \hline z_2 \end{array}\right)}_{\left(\begin{array}{c} \zeta_1 \\ \hline l_{21}\zeta_1 + L_{22}z_2 \end{array}\right)} = \left(\begin{array}{c} \beta_1 \\ \hline b_2 \end{array}\right)$$

$$\left(\begin{array}{c} \zeta_1 = \beta_1 \\ \hline L_{22}z_2 = -l_{21}\zeta_1 + b_2 \end{array}\right)$$

Consider

$$Lz = b$$

$$\left(\begin{array}{c} \zeta_1 = \beta_1 \\ \hline L_{22}z_2 = -l_{21}\zeta_1 + b_2 \end{array}\right)$$

$$\left(\begin{array}{c} \zeta_1 := \beta_1 \\ \hline b_2 := -\zeta_1 l_{21} + b_2 \end{array}\right)$$

$L_{22}z_2 = b_2$

axpy

→ scalar로 만들기 위해서 b2를 왼쪽 변으로 빼냈다.

- 질문!!! 요게 무슨말인가?? 빼면 되는 것인가?? 무슨 말이지??? L22z2=b2. 이 식은 어디서 나온거지?? 아직도 이해를 못하겠다. ㅎㅎㅎㅎ

- 할튼, 알고리즘을 정리하면 (윗부분이 이해가 가야 다음이 진행될 것이다.

Algorithm: $[b] := \text{LTRSV\_UNB\_VAR1}(L, b)$

Partition $L \rightarrow \left(\begin{array}{c|c} L_{TL} & 0 \\ \hline L_{BL} & L_{BR} \end{array}\right)$, $b \rightarrow \left(\begin{array}{c} b_T \\ \hline b_B \end{array}\right)$

where L_{TL} is 0×0, b_T has 0 rows

while $m(L_{TL}) < m(L)$ **do**

Repartition

$$\left(\begin{array}{c|c} L_{TL} & 0 \\ \hline L_{BL} & L_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} L_{00} & 0 & 0 \\ \hline l_{10}^T & \lambda_{11} & 0 \\ \hline L_{20} & l_{21} & L_{22} \end{array}\right), \left(\begin{array}{c} b_T \\ \hline b_B \end{array}\right) \rightarrow \left(\begin{array}{c} b_0 \\ \hline \beta_1 \\ \hline b_2 \end{array}\right)$$

$b_2 := b_2 - \beta_1 l_{21}$ axpy

Continue with

$$\left(\begin{array}{c|c} L_{TL} & 0 \\ \hline L_{BL} & L_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} L_{00} & 0 & 0 \\ \hline l_{10}^T & \lambda_{11} & 0 \\ \hline L_{20} & l_{21} & L_{22} \end{array}\right), \left(\begin{array}{c} b_T \\ \hline b_B \end{array}\right) \leftarrow \left(\begin{array}{c} b_0 \\ \hline \beta_1 \\ \hline b_2 \end{array}\right)$$

endwhile

Algorithm: $b := \text{FORWARD\_SUBSTITUTION}(A, b)$

Partition $A \rightarrow \left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right)$, $b \rightarrow \left(\begin{array}{c} b_T \\ \hline b_B \end{array}\right)$

where A_{TL} is 0×0, b_T has 0 rows

while $m(A_{TL}) < m(A)$ **do**

Repartition

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right), \left(\begin{array}{c} b_T \\ \hline b_B \end{array}\right) \rightarrow \left(\begin{array}{c} b_0 \\ \hline \beta_1 \\ \hline b_2 \end{array}\right)$$

$b_2 := b_2 - \beta_1 a_{21}$ $\quad (= b_2 - \beta_1 l_{21})$

Continue with

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right), \left(\begin{array}{c} b_T \\ \hline b_B \end{array}\right) \leftarrow \left(\begin{array}{c} b_0 \\ \hline \beta_1 \\ \hline b_2 \end{array}\right)$$

endwhile

→ lower triangle Lz=b를 푸는 과정과 forward substitution이랑 같은 것이다.

(3) Ux=b를 푸는 것은 Backward substitution이다.

Partition

$$U \rightarrow \left(\begin{array}{c|c} \upsilon_{11} & u_{12}^T \\ \hline 0 & U_{22} \end{array}\right), x \rightarrow \left(\begin{array}{c} \chi_1 \\ \hline x_2 \end{array}\right), b \rightarrow \left(\begin{array}{c} \beta_1 \\ \hline b_2 \end{array}\right)$$

Consider

$$Ux = b$$

Substitute:

$$\underbrace{\left(\begin{array}{c|c} \upsilon_{11} & u_{12}^T \\ \hline 0 & U_{22} \end{array}\right)\left(\begin{array}{c} \chi_1 \\ \hline x_2 \end{array}\right)}_{\left(\begin{array}{c} \upsilon_{11}\chi_1 + u_{12}^T x_2 \\ \hline U_{22}x_2 \end{array}\right)} = \left(\begin{array}{c} \beta_1 \\ \hline b_2 \end{array}\right)$$

$$\left(\begin{array}{c} \upsilon_{11}\chi_1 + u_{12}^T x_2 = \beta_1 \\ \hline U_{22}x_2 = b_2 \end{array}\right)$$

→ partition을 하면 풀 수 있다.

Partition

$$U \rightarrow \left(\begin{array}{c|c} \upsilon_{11} & u_{12}^T \\ \hline 0 & U_{22} \end{array}\right), x \rightarrow \left(\begin{array}{c} \chi_1 \\ \hline x_2 \end{array}\right) b \rightarrow \left(\begin{array}{c} \beta_1 \\ \hline b_2 \end{array}\right)$$

Consider

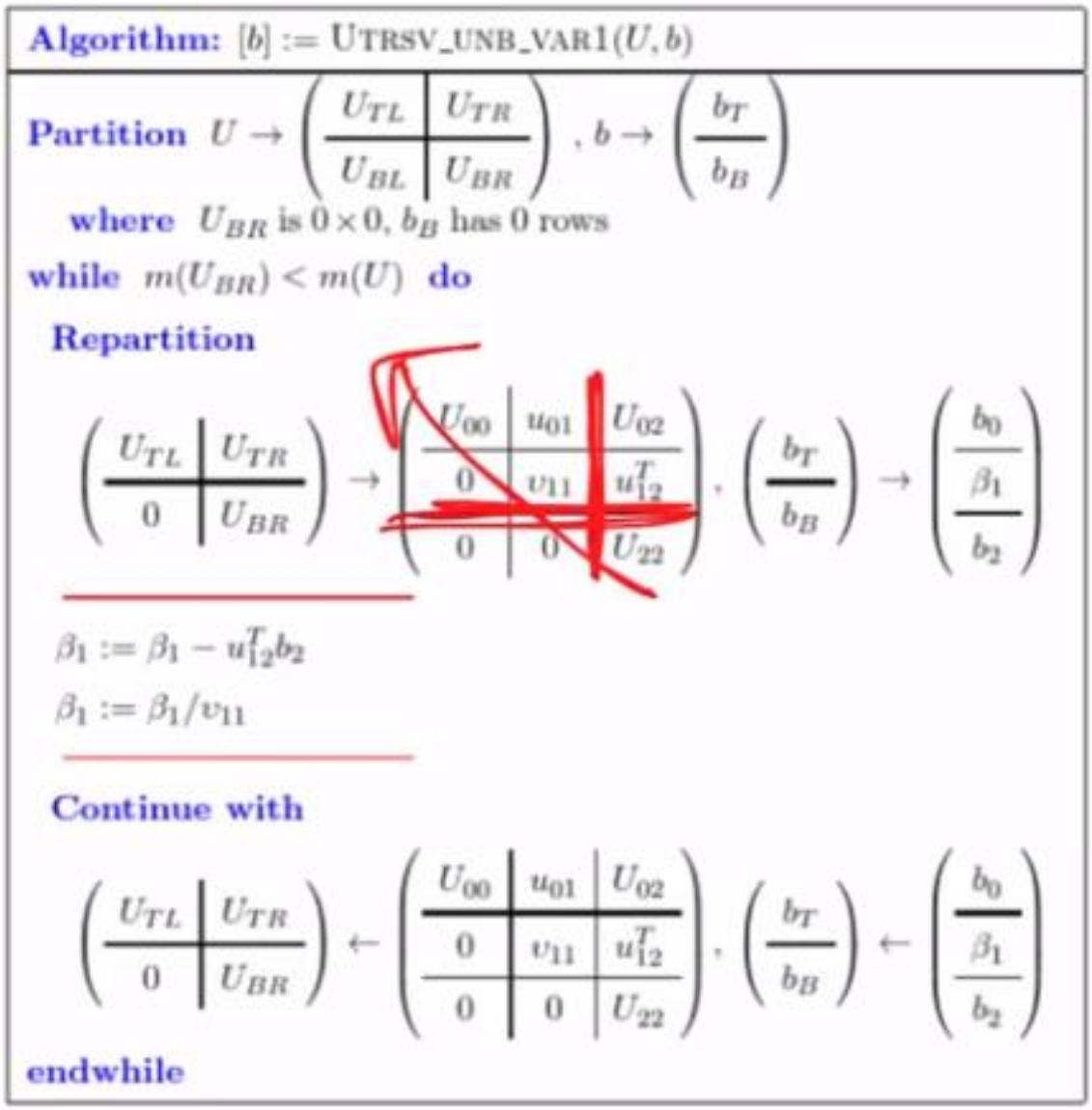
$$Ux = b$$

$$\left(\begin{array}{c} \upsilon_{11}\chi_1 + u_{12}^T x_2 = \beta_1 \\ \hline U_{22}x_2 = b_2 \end{array}\right)$$

$$\left(\begin{array}{c} \chi_1 := (\beta_1 - u_{12}^T x_2)/\upsilon_{11} \\ \hline b_2 = x_2 \end{array}\right)$$

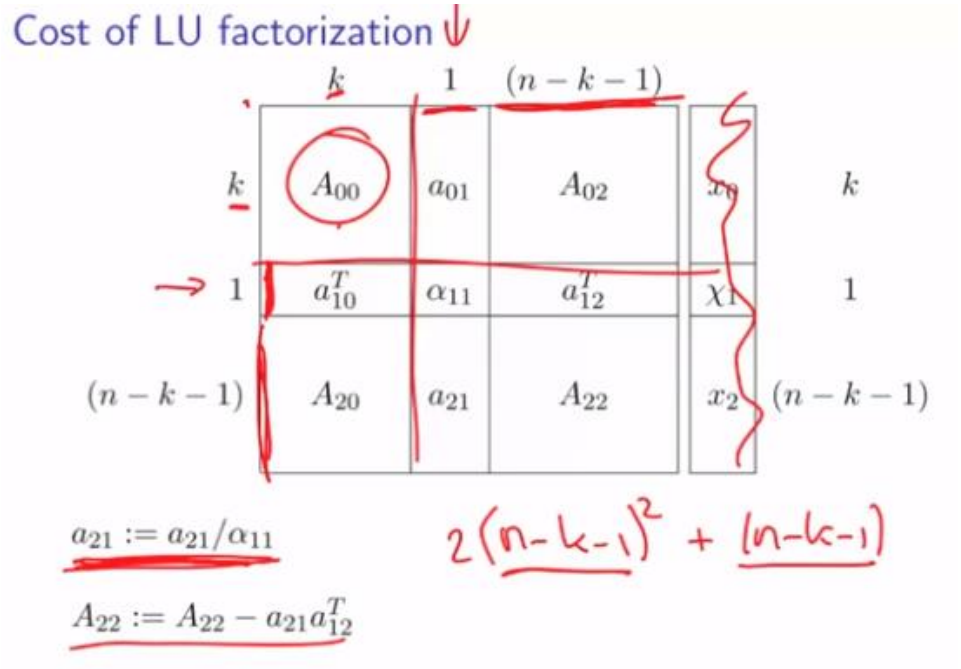
→ x2를 안다고 해보자. x2가 overwritten b2라고 해보자. 질문 : 와 왜 b2가 x2가 되는 것인가???

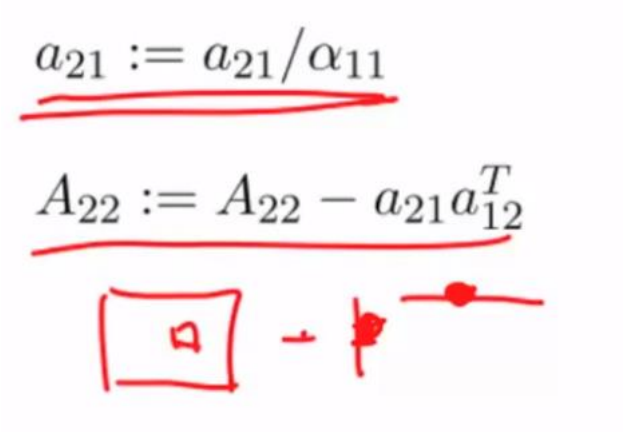
- 알고리즘



6.4. Cost

6.4.1. Cost of LU factorization





→ rank-1 update하는데에 있어서 2번 연산을 해야한다. 곱하고 빼고. 이것을 각 행렬에 해야 하니까 2(n-k-1)^2의 cost가 필요하다. 그런데 n-k-1은 무시할 수 있다.

→ 총 cost를 구하려면 k가 0부터 n-1까지 가는데 각각 하기 때문에

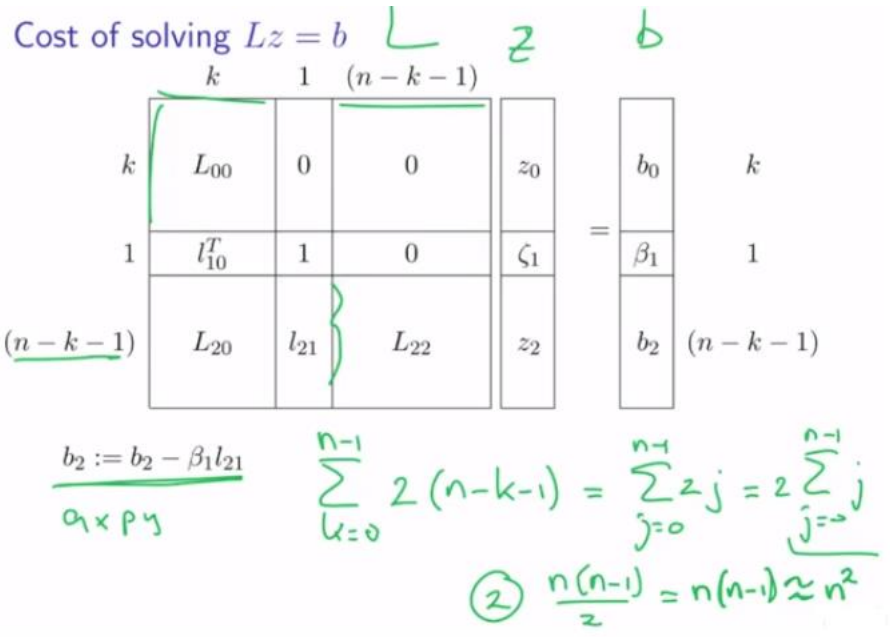
$$\sum_{k=0}^{n-1} 2(n-k-1)^2$$

$$k = 0, 1, 2, \cdots, n-1$$
$$n-k-1 = n-1, n-2, \cdots, 0$$

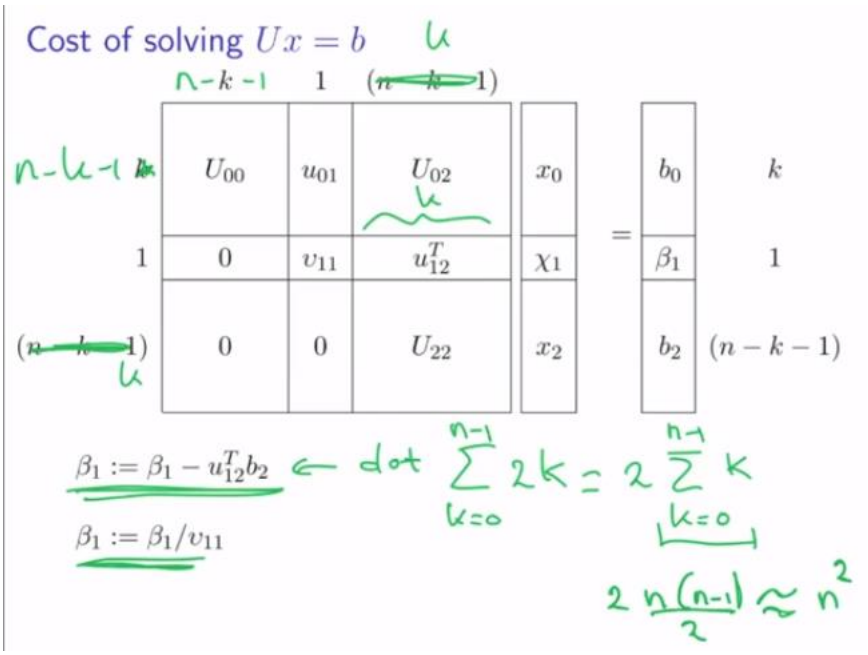
$$2\sum_{k=0}^{n-1} (n-k-1)^2 = 2\sum_{j=0}^{n-1} j^2$$

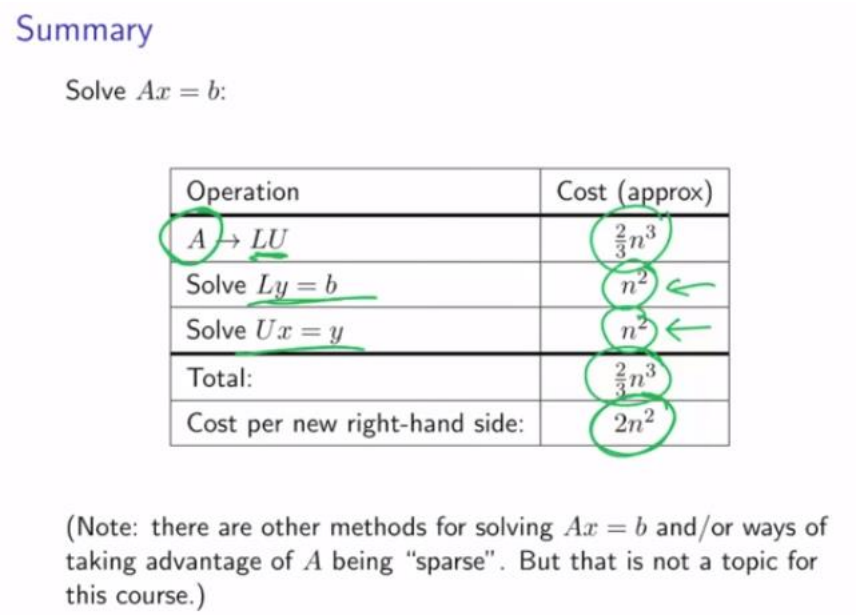
$$2\frac{(n-1)n(2n-1)}{6}$$

6.4.2. lower triangle을 푸는데(Ly=b) 드는 cost



6.4.3. upper triangle을 푸는데 드는 cost





Summary

Solve $Ax = b$:

Operation	Cost (approx)
$A \rightarrow LU$	$\frac{2}{3}n^3$
Solve $Ly = b$	n^2
Solve $Ux = y$	n^2
Total:	$\frac{2}{3}n^3$
Cost per new right-hand side:	$2n^2$

(Note: there are other methods for solving $Ax = b$ and/or ways of taking advantage of A being "sparse". But that is not a topic for this course.)

6.5. Blocked LU factorization

What you saw in Week 5, Units 5.4.1 and 5.4.2, was that by carefully implementing matrix-matrix multiplication, the performance of this operation could be improved from a few percent of the peak of a processor to better than 90%.

---

7. Advanced LU factorization

When Gaussian Elimination Breaks Down

7.2.1. When Gaussian Elimination Works

7.2.2. The Problem

7.2.3. Permutations

7.2.4. Gaussian Elimination with Row Swapping (LU factorization with Partial Pivoting)

7.2.5. When Gaussian Elimination Fails Altogether

The Inverse Matrix

7.3.1. Inverse Functions in 1D

7.3.2. Back to Linear Transformations

7.3.3. Simple Examples

7.3.4. More Advanced (but Still Simple) Examples

7.3.5. Properties

Enrichment

7.4.1. Blocked Algorithms for LU with Partial Pivoting

7.4.2. Library Routines for LU with Partial Pivoting

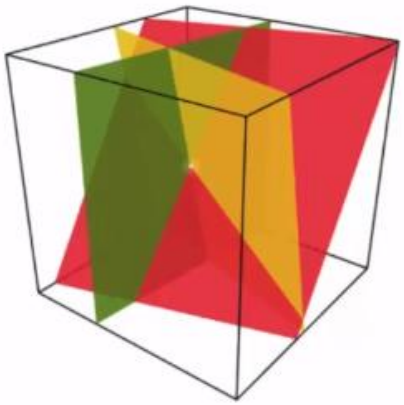
7.1. Advanced LU factorization

Ax=b는 해가 1개 있을 수도 있고, 무한 개일 수도 있고, 없을 수도 있다.

만약에 1개의 해만 있다고 하면 역행렬을 구하면 된다. LU factorization 과정에서 0으로 나누는 경우가 발생하기 때문에 이를 생각해본다.

Interpreting Multiple Linear Equations

$$\begin{aligned} \alpha_{0,0}\chi_0 + \alpha_{0,1}\chi_1 + \alpha_{0,2}\chi_2 &= \beta_0 \\ \alpha_{1,0}\chi_0 + \alpha_{1,1}\chi_1 + \alpha_{1,2}\chi_2 &= \beta_1 \\ \alpha_{2,0}\chi_0 + \alpha_{2,1}\chi_1 + \alpha_{2,2}\chi_2 &= \beta_2 \end{aligned}$$



(image from Wikipedia)

< 큰 그림 요약 >

Now, we can see that when executing Gaussian elimination (LU factorization) with Ax=b where A is a square matrix, one of three things can happen:

(1) 가능성 1. LU factorization이 성공적으로 되고 U 대각선에 0이 없는 경우 ➔ 유일한 해

The process completes with no zeroes on the diagonal of the resulting matrix U. Then A=LU and Ax=b has a unique solution, which can be found by solving Lz=b followed by Ux=z.

(2) 가능성 2. Row를 바꿔야하는 경우에 ➔ Lz=Pb를 한다(**Permutation**). ➔ 유일한 해를 가진다.

The process requires row exchanges, completing with no zeroes on the diagonal of the resulting matrix U. Then PA=LU and Ax=b has a unique solution, which can be found by solving Lz=Pb followed by Ux=z.

(3) 가능성 3. Pivoting을 하더라도 대각선에 0이 생길 수 밖에 없는 경우이다. ➔ 해가 없거나(불능이거나) 부정이다.

The process requires row exchanges, but at some point no row can be found that puts a nonzero on the diagonal, at which point the process fails (unless the zero appears as the last element on the diagonal, in which case it completes, but leaves a zero on the diagonal).

7.1. LU factorization이 잘 되고 해가 유일한 경우

7.1.1. 수학적 명제

Summary

Let $A \in \mathbb{R}^{n \times n}$ and assume that $A \to LU$ completes with a matrix U that has no zero elements on its diagonal.

$Ax = b$ has a unique solution.

Always/Sometimes/Never

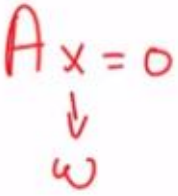
→ U행렬이 대각선에 0이 없고 LU factorization이 성공적으로 잘 되면, lower triangle (=forward substitution)구하고, upper triangle (Ux=b)를 구하는 과정(=backward substitution을 하면 유일한 해를 구할 수 있다.

7.1.2. 수학적 증명

→ 그런데 이게 과정 유일한 해일까?

- 두 개의 해가 있다고 해보자. 두개의 해는 u, v라고 하면 A(u-v)=0이 될 것이다. 만약에 u-v=x일 때, Ax=0을 만족하는 x가 0이 아니어야 두 개의 해가 있게 된다. X=0만 이 식을 만족시킨다면 해는 유일한 해가 된다.

- Assume $Ax = b$ has two solutions, call them u and v.
- Then $Au = b$ and $Av = b$
- Hence $A(u - v) = Au - Av = b - b = 0$.



→ 이를 수학적으로 풀려면 A=LU를 해보면 된다. (LU factorization이 성공적으로 진행됐다는 가정이 있으므로, L과 U가 잘 정의된다.)

- Then $(LU)w = 0$
- or $L(Uw) = 0$

What we will show is that $Lz = 0$ implies that $z = 0$ and $Uw = 0$ implies $w = 0$ and hence $u = v$.

- or $L(Uw) = 0$

What we will show is that $Lz = 0$ implies that $z = 0$ and $Uw = 0$ implies $w = 0$ and hence $u = v$.

(1) Lz=0이면 z=0이다? 이거를 증명해보자.

$Lz = 0$ implies $z = 0$

$$\begin{pmatrix} 1 & 0 & 0 & \cdots & 0 \\ \lambda_{1,0} & 1 & 0 & \cdots & 0 \\ \lambda_{2,0} & \lambda_{2,1} & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \\ \lambda_{n-1,0} & \lambda_{n-1,1} & \lambda_{n-1,2} & \cdots & 1 \end{pmatrix} \begin{pmatrix} \zeta_0 \\ \zeta_1 \\ \zeta_2 \\ \vdots \\ \zeta_{n-1} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix}$$

→ L은 unit triangular matrix라고 해보자.

$$\begin{pmatrix} 1 & 0 & 0 & \cdots & 0 \\ \lambda_{1,0} & 1 & 0 & \cdots & 0 \\ \lambda_{2,0} & \lambda_{2,1} & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \\ \lambda_{n-1,0} & \lambda_{n-1,1} & \lambda_{n-1,2} & \cdots & 1 \end{pmatrix} \begin{pmatrix} \zeta_0 \\ \zeta_1 \\ \zeta_2 \\ \vdots \\ \zeta_{n-1} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix}$$

$\zeta_0 = 0$ $\zeta_1 = 0$

(2) Uw=0 implies w=0인가???

U의 대각선이 0이 아닌 것을 알기 때문에

$Uw = 0$ implies $w = 0$

$$\begin{pmatrix} \upsilon_{0,0} & \cdots & \upsilon_{0,n-3} & \upsilon_{0,n-2} & \upsilon_{0,n-1} \\ \vdots & \ddots & \vdots & \vdots & \\ 0 & \cdots & \upsilon_{n-3,n-3} & \upsilon_{n-3,n-2} & \upsilon_{n-3,n-1} \\ 0 & \cdots & 0 & \upsilon_{n-2,n-2} & \upsilon_{n-2,n-1} \\ 0 & \cdots & 0 & 0 & \upsilon_{n-1,n-1} \end{pmatrix} \begin{pmatrix} \omega_0 \\ \vdots \\ \omega_{n-3} \\ \omega_{n-2} \\ \omega_{n-1} \end{pmatrix} = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

→ w가 0이다.

7.2. 문제 : 유일한 해가 있으나 알고리즘이 해를 못 구하는 경우(When Gaussian elimination fails)

Gauss elimination 혹은 LU factorization이 안 되는 경우 : 0으로 나누는 경우와 맞닥뜨릴 수 있다.

Repartition

$$\begin{pmatrix} A_{TL} & A_{TR} \\ A_{BL} & A_{BR} \end{pmatrix} \rightarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right)$$

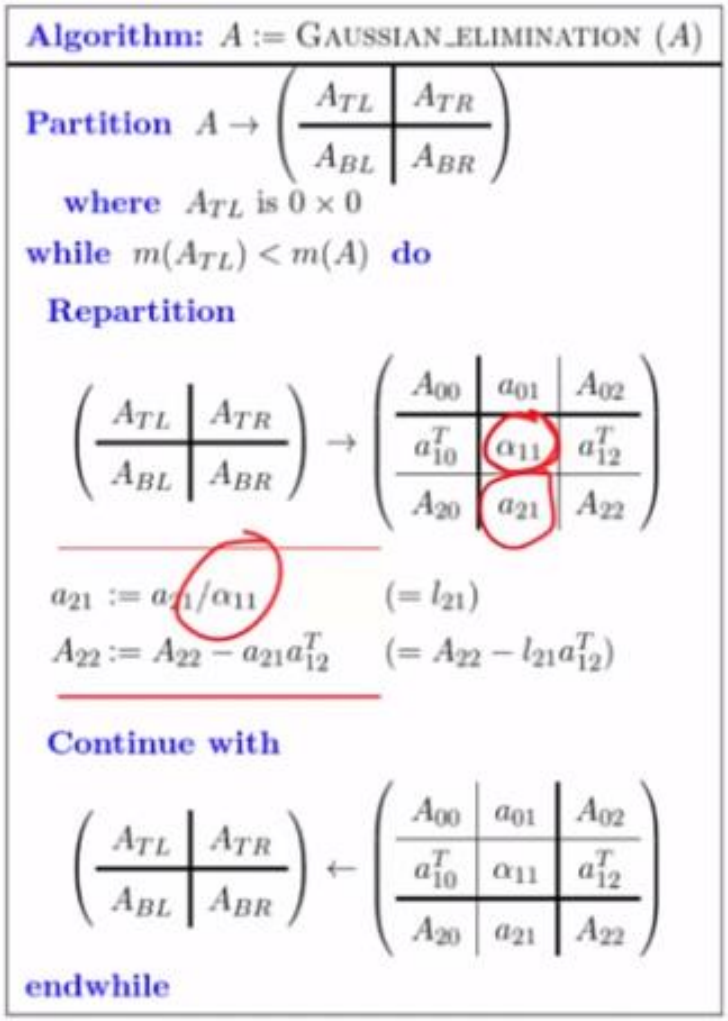
$a_{21} := a_{21}/\alpha_{11}$ $(= l_{21})$

$A_{22} := A_{22} - a_{21}a_{12}^T$ $(= A_{22} - l_{21}a_{12}^T)$

Continue with

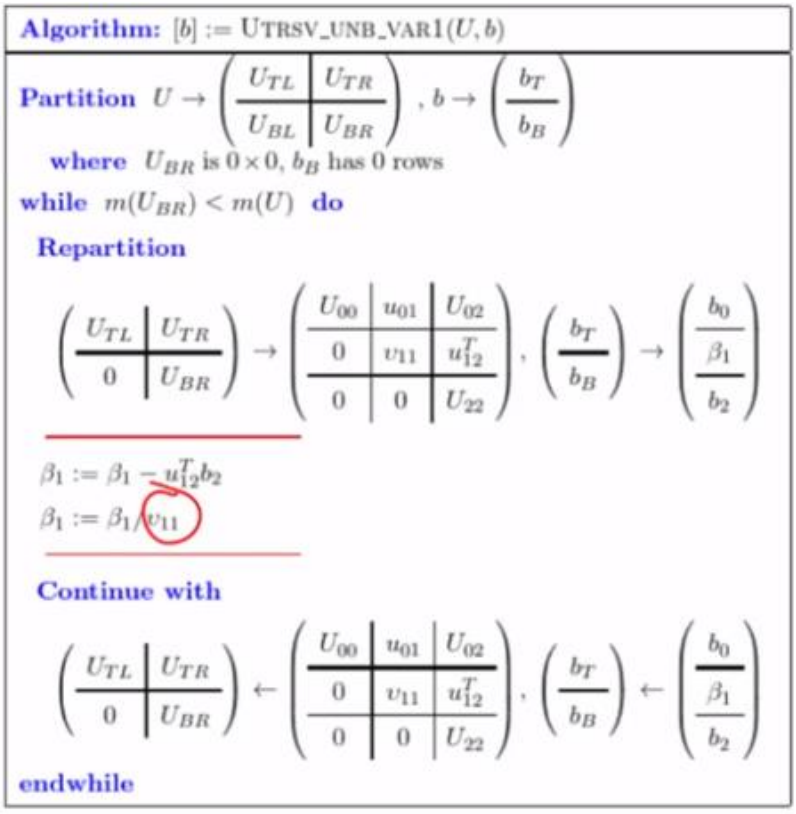
$$\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$

(1) Gaussian elimination 과정에서



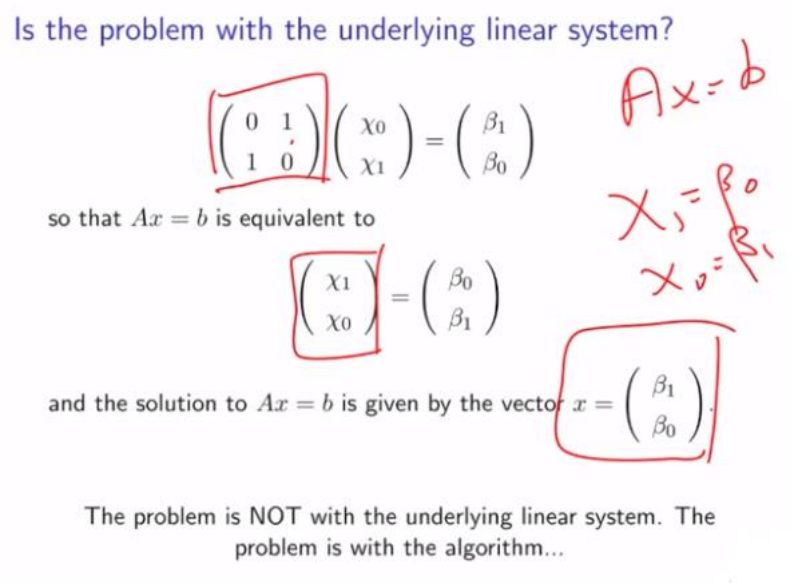
→ 여기서 만약에 0으로 나눠야 하는 순간이 언젠가 중간에 오게 되면 안 된다.

(2) LU factorization 과정에서



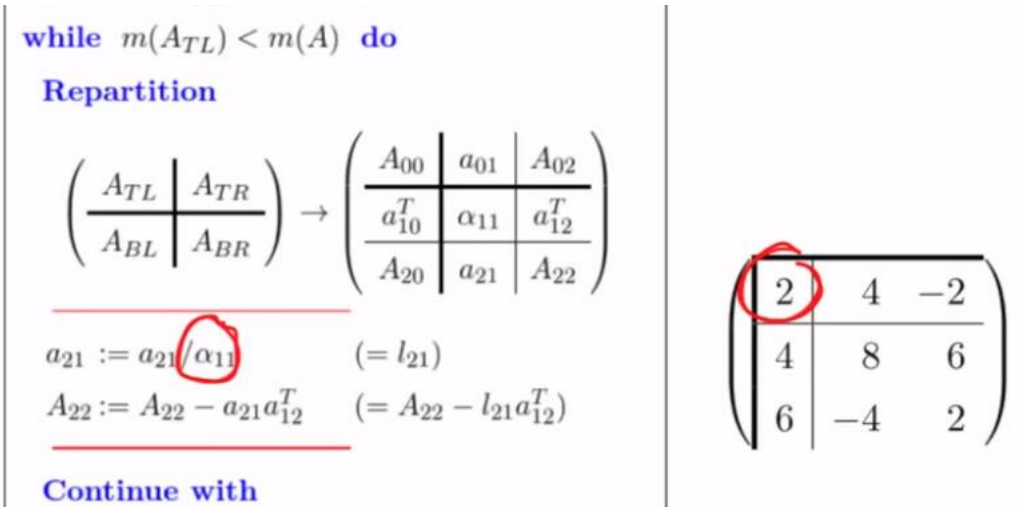
→ 대각선에 0이 오면 안된다.

- 예시1 : Ax=B에서 해가 1개인 A ($\begin{matrix} 0 & 1 \\ 1 & 0 \end{matrix}$)를 대입해보면, 해는 있으나, 알고리즘이 작동을 잘 못한다는 것을 알 수 있다. 대각선에 0이 있기 때문이다.



→ 물론 위의 식에서 beta_0과 1의 자리가 바뀌었지만(오타) 어쨌든 간에 algorithm에 문제가 있다는 것을 알 수 있다. Matrix(linear system)에 문제가 있지 않다는 것을 알 수 있다!

- 예시 2 :



while $m(A_{TL}) < m(A)$ **do**

Repartition

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right)$$

$a_{21} := a_{21}/\alpha_{11}$ $\quad (= l_{21})$

$A_{22} := A_{22} - a_{21}a_{12}^T$ $\quad (= A_{22} - l_{21}a_{12}^T)$

Continue with

$$\left(\begin{array}{c|c|c} 2 & 4 & -2 \\ \hline 2 & 0 & 10 \\ \hline 3 & -16/0 & 8 \end{array}\right)$$

7.3. 문제의 해결 : Permutation

7.3.1. Permutation matrix과 Pivot matrix

- partitioning의 응용

$$\left(\begin{array}{c|c|c} 0 & 1 & 0 \\ \hline 0 & 0 & 1 \\ \hline 1 & 0 & 0 \end{array}\right)\left(\begin{array}{c} -2\ \ 1\ \ 2 \\ \hline 3\ \ 2\ \ 1 \\ \hline -1\ \ 0\ \ -3 \end{array}\right)$$

$$= \left(\begin{array}{c} 0\times\left(-2\ \ 1\ \ 2\right)+1\times\left(3\ \ 2\ \ 1\right)+0\times\left(-1\ \ 0\ \ -3\right) \\ \hline 0\times\left(-2\ \ 1\ \ 2\right)+0\times\left(3\ \ 2\ \ 1\right)+1\times\left(-1\ \ 0\ \ -3\right) \\ \hline 1\times\left(-2\ \ 1\ \ 2\right)+0\times\left(3\ \ 2\ \ 1\right)+0\times\left(-1\ \ 0\ \ -3\right) \end{array}\right)$$

$$= \left(\begin{array}{c} 3\ \ 2\ \ 1 \\ \hline -1\ \ 0\ \ -3 \\ \hline -2\ \ 1\ \ 2 \end{array}\right).$$

→ 보니까 행렬을 막 위아래로 섞어놓는 효과가 있다. 이 행렬을 permutation matrix라고 한다.

Another way:

$$\left(\begin{array}{c|c|c} 0 & 1 & 0 \\ \hline 0 & 0 & 1 \\ \hline 1 & 0 & 0 \end{array}\right)\left(\begin{array}{c} -2\ \ 1\ \ 2 \\ \hline 3\ \ 2\ \ 1 \\ \hline -1\ \ 0\ \ -3 \end{array}\right) = \left(\begin{array}{c} e_1^T \\ \hline e_2^T \\ \hline e_0^T \end{array}\right)\left(\begin{array}{c} -2\ \ 1\ \ 2 \\ 3\ \ 2\ \ 1 \\ -1\ \ 0\ \ -3 \end{array}\right)$$

$$= \left(\begin{array}{c} e_1^T\left(\begin{array}{c} -2\ \ 1\ \ 2 \\ 3\ \ 2\ \ 1 \\ -1\ \ 0\ \ -3 \end{array}\right) \\ \hline e_2^T\left(\begin{array}{c} -2\ \ 1\ \ 2 \\ 3\ \ 2\ \ 1 \\ -1\ \ 0\ \ -3 \end{array}\right) \\ \hline e_0^T\left(\begin{array}{c} -2\ \ 1\ \ 2 \\ 3\ \ 2\ \ 1 \\ -1\ \ 0\ \ -3 \end{array}\right) \end{array}\right) =$$

→ 이 방법으로 보면 더 일반화를 할 수 있게 한다. E1T는 2번째 행을 다오.

- permutation matrix 정의

Definition

A vector with integer components

$$p = \begin{pmatrix} k_0 \\ k_1 \\ \vdots \\ k_{n-1} \end{pmatrix}$$

is said to be a permutation vector if

- $k_j \in \{0, \ldots, n-1\}$, for $0 \leq j < n$; and
- $k_i = k_j$ implies $i = j$.

We will often write $(k_0, k_1, \ldots, k_{n-1})^T$.

Examples: $(0,1,2,3)^T$, $(1,2,3,0)^T$, $(0,3,2,1)^T$.
Not an example: $(0,0,3,2)^T$.

Definition

Let $p = (k_0, \ldots, k_{n-1})^T$ be a permutation vector. Then

$$P = P(p) = \begin{pmatrix} e_{k_0}^T \\ e_{k_1}^T \\ \vdots \\ e_{k_{n-1}}^T \end{pmatrix}$$

is said to be a *permutation matrix*.

→ 다시 unit basis vector가 된 것이다.

Let $p = (k_0, \ldots, k_{n-1})^T$ be a permutation.

$$Px = P(p)x = \begin{pmatrix} e_{k_0}^T \\ \hline e_{k_1}^T \\ \hline \vdots \\ \hline e_{k_{n-1}}^T \end{pmatrix} x = \begin{pmatrix} e_{k_0}^T x \\ \hline e_{k_1}^T x \\ \hline \vdots \\ \hline e_{k_{n-1}}^T x \end{pmatrix} = \begin{pmatrix} \chi_{k_0} \\ \hline \chi_{k_1} \\ \hline \vdots \\ \hline \chi_{k_{n-1}} \end{pmatrix}.$$

- x를 행렬A로 바꾸면, 행이 reshuffle이 된다.

Let $p = (k_0, \ldots, k_{n-1})^T$ be a permutation.

$$A = \left(\begin{array}{c} \widetilde{a}_0^T \\ \hline \widetilde{a}_1^T \\ \hline \vdots \\ \hline \widetilde{a}_{n-1}^T \end{array} \right)$$

$$PA = P(p)A = \left(\begin{array}{c} e_{k_0}^T \\ \hline e_{k_1}^T \\ \hline \vdots \\ \hline e_{k_{n-1}}^T \end{array} \right) A = \left(\begin{array}{c} e_{k_0}^T A \\ \hline e_{k_1}^T A \\ \hline \vdots \\ \hline e_{k_{n-1}}^T A \end{array} \right) = \left(\begin{array}{c} \widetilde{a}_{k_0}^T \\ \hline \widetilde{a}_{k_1}^T \\ \hline \vdots \\ \hline \widetilde{a}_{k_{n-1}}^T \end{array} \right).$$

- 오른쪽에서 permutation된 거를 transpose에서 곱하게 되면

$$A = \left(\begin{array}{c|c|c|c} a_0 & a_1 & \cdots & a_{n-1} \end{array} \right)$$

$$\begin{aligned} AP^T &= A \left(\begin{array}{c} e_{k_0}^T \\ e_{k_1}^T \\ \vdots \\ e_{k_{n-1}}^T \end{array} \right)^T = A \left(\begin{array}{c|c|c|c} e_{k_0} & e_{k_1} & \cdots & e_{k_{n-1}} \end{array} \right) \\ &= \left(\begin{array}{c|c|c|c} Ae_{k_0} & Ae_{k_1} & \cdots & Ae_{k_{n-1}} \end{array} \right) \\ &= \left(\begin{array}{c|c|c|c} a_{k_0} & a_{k_1} & \cdots & a_{k_{n-1}} \end{array} \right). \end{aligned}$$

- **pivot matrix**는 permutation matrix에서 0행이랑 파이 행이랑 바꾼 것이다.

Pivot matrix:

$$\widetilde{P}(\pi) = \left(\begin{array}{c} e_\pi^T \\ e_1^T \\ \vdots \\ e_{\pi-1}^T \\ e_0^T \\ e_{\pi+1}^T \\ \vdots \\ e_{n-1}^T \end{array} \right) = \left(\begin{array}{c|ccc|c|ccc} 0 & 0 & \cdots & 0 & 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 & 0 & 0 & \cdots & 0 \\ \hline 1 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 \\ \hline 0 & 0 & \cdots & 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & 1 \end{array} \right)$$

$$A = \left(\begin{array}{c|c|c|c} a_0 & a_1 & \cdots & a_{n-1} \end{array} \right)$$

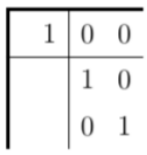
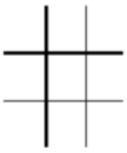
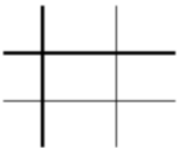
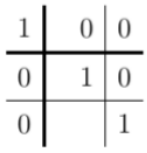
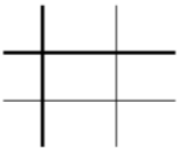
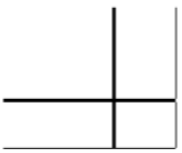
$$AP(\pi)^T = AP(\pi) = A \begin{pmatrix} \boxed{e_\pi^T} \\ e_1^T \\ \vdots \\ e_{\pi-1}^T \\ \boxed{e_0^T} \\ e_{\pi+1}^T \\ \vdots \\ e_{n-1}^T \end{pmatrix}^T$$

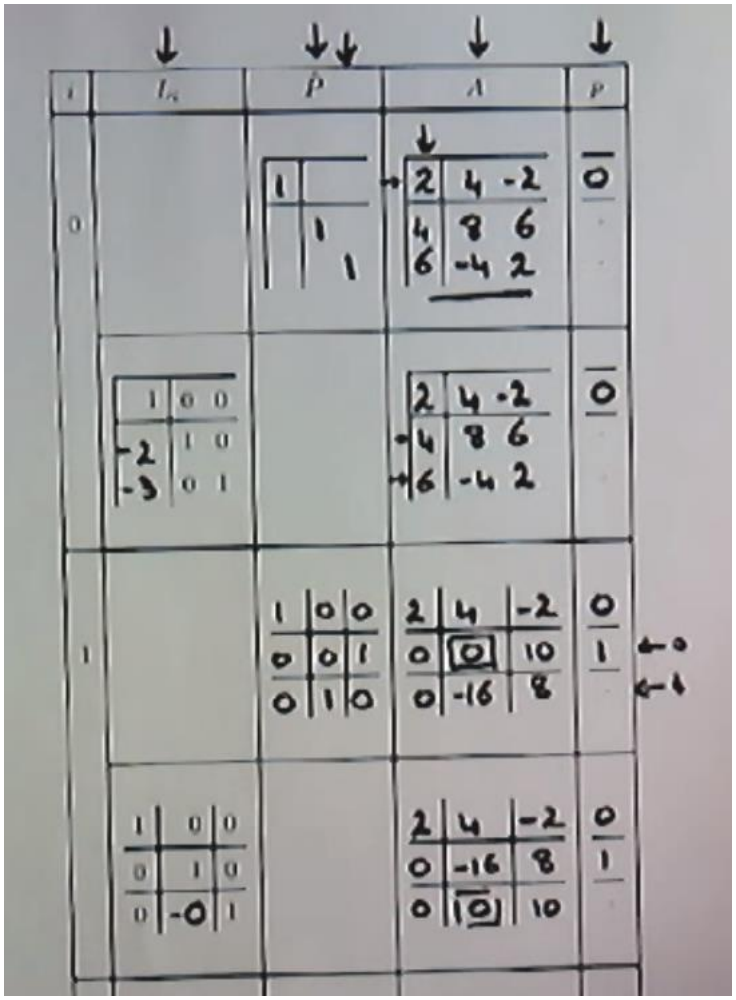
$$= A \left(\begin{array}{c|c|c|c|c|c|c|c} \boxed{e_\pi} & e_1 & \cdots & e_{\pi-1} & \boxed{e_0} & e_{\pi+1} & \cdots & e_{n-1} \end{array} \right)$$

$$= \left(\begin{array}{c|c|c|c|c|c|c|c} \boxed{a_\pi} & a_1 & \cdots & a_{\pi-1} & \boxed{a_0} & a_{\pi+1} & \cdots & a_{n-1} \end{array} \right)$$

- 요약 : permutation from the right swipe columns(열) / permutation from the left swipe rows(행).

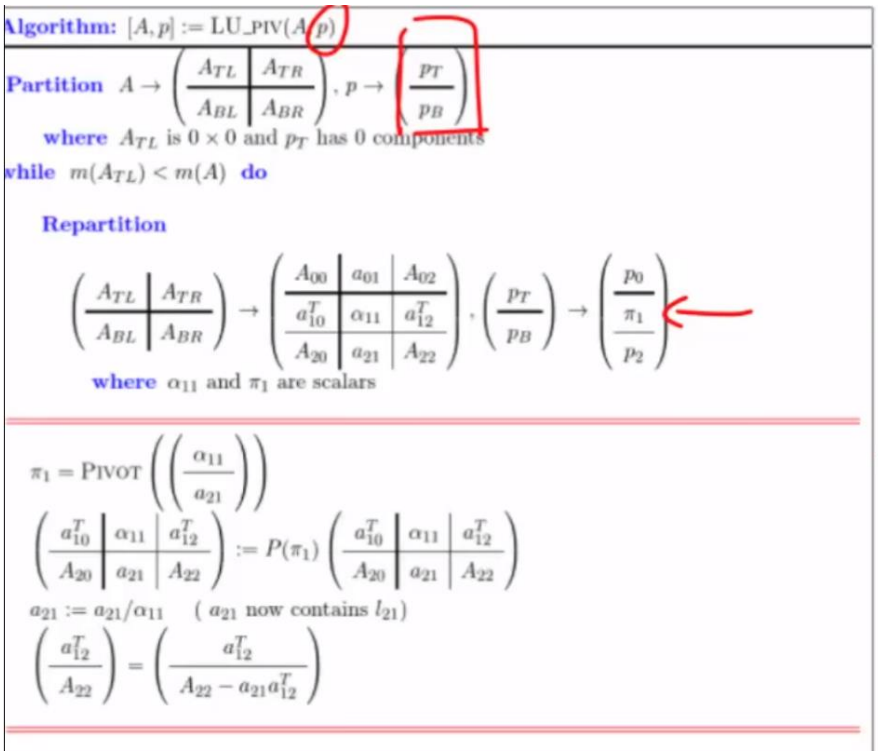
7.3.2. Gaussian Elimination with Row Swapping (LU Factorization with Partial Pivoting)

i	L_i	$\tilde{P}$	A	p
0				
				
1				
				
2				



→ 저장된 vector도 swiping된다는 것을 잊어서는 안 된다!

- 알고리즘을 짜보자.



Algorithm: $[A, p] := \text{LU\_PIV}(A, p)$

Partition $A \rightarrow \left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right)$, $p \rightarrow \left(\begin{array}{c} p_T \\ \hline p_B \end{array}\right)$

where A_{TL} is 0×0 and p_T has 0 components

while $m(A_{TL}) < m(A)$ **do**

Repartition

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right), \left(\begin{array}{c} p_T \\ \hline p_B \end{array}\right) \rightarrow \left(\begin{array}{c} p_0 \\ \hline \pi_1 \\ \hline p_2 \end{array}\right)$$

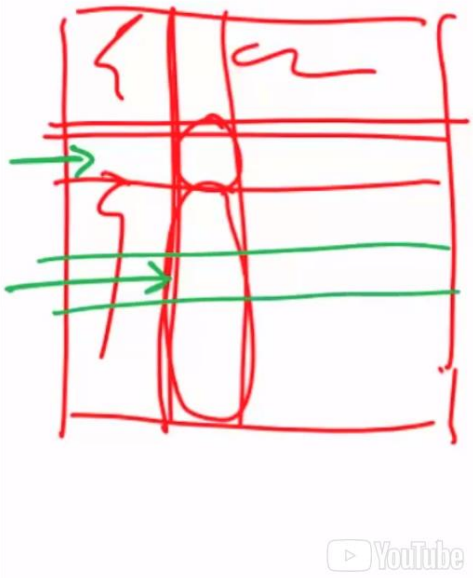
where α_{11} and π_1 are scalars

$$\pi_1 = \text{PIVOT}\left(\left(\begin{array}{c} \alpha_{11} \\ \hline a_{21} \end{array}\right)\right)$$

$$\left(\begin{array}{c|c|c} a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right) := P(\pi_1) \left(\begin{array}{c|c|c} a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right)$$

$a_{21} := a_{21}/\alpha_{11}$ (a_{21} now contains l_{21})

$$\left(\begin{array}{c} a_{12}^T \\ \hline A_{22} \end{array}\right) = \left(\begin{array}{c} a_{12}^T \\ \hline A_{22} - a_{21}a_{12}^T \end{array}\right)$$



→ 0이 대각선에 나오면, 그 열에서 0이 아닌 애와 swapping을 해야 한다. 즉 pivot matrix를 통해서 permutation을 하면 된다.

- 아래와 같이 permutation을 하게 된다.

Recap:

$$\left(\begin{array}{c|cc} 1 & 0 & 0 \\ \hline 0 & 1 & 0 \\ 0 & -0 & 1 \end{array}\right) \times \left(\begin{array}{c|cc} 0 & 0 & 0 \\ \hline 0 & 0 & 1 \\ 0 & 1 & 0 \end{array}\right)$$

$$\times \left(\begin{array}{c|cc} 1 & 0 & 0 \\ \hline -2 & 1 & 0 \\ -3 & 0 & 1 \end{array}\right) \times \left(\begin{array}{c|cc} 1 & 0 & 0 \\ \hline 0 & 1 & 0 \\ 0 & 0 & 1 \end{array}\right) \times \begin{pmatrix} 2 & 4 & -2 \\ 4 & 8 & 6 \\ 6 & -4 & 2 \end{pmatrix}$$

$$= \begin{pmatrix} 2 & 4 & 2 \\ 3 & -16 & 8 \\ 2 & 0 & 10 \end{pmatrix} = L\,U = P\,A$$

→ 이 permutation은 A에 반영되어야 한다.

More generally...

$$\tilde{L}_{n-2}P_{n-2}\cdots\tilde{L}_1P_1\tilde{L}_0P_0A = U$$

yields the same result as

$$\underbrace{P_{n-2}\cdots P_1P_0}_{P}\,A = LU$$

Solving $Ax = b$:

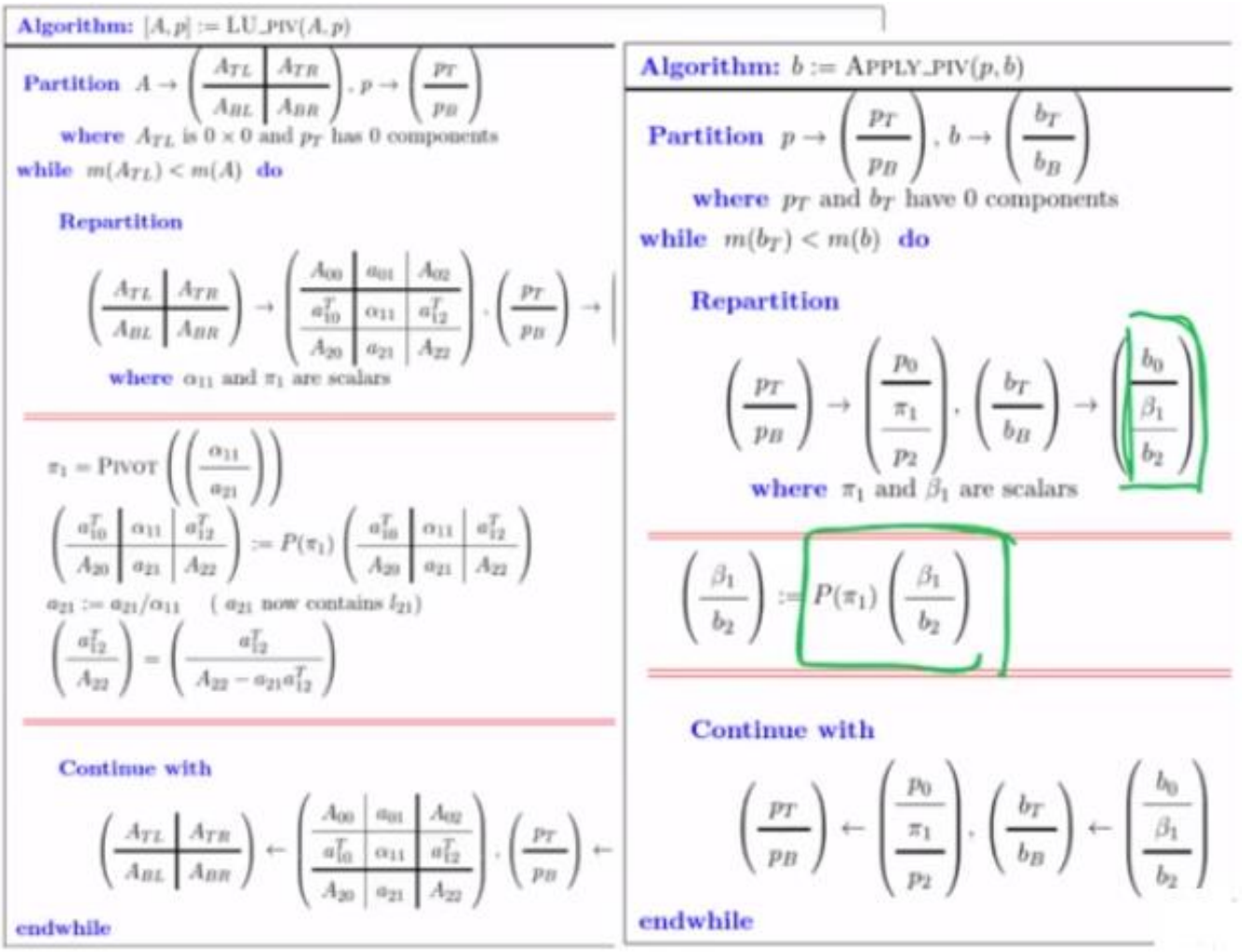
$$\underbrace{P_{n-2}\cdots P_1P_0}_{P}\,Ax = \underbrace{P_{n-2}\cdots P_1P_0}_{P}\,b$$

$$\underbrace{LU}$$

or

$$L\,\underbrace{(Ux)}_{z} = P_{n-2}\cdots P_1P_0b$$

- 또한 right hand side vector도 permutation해야 한다.



- LU factorization에 몇 가지 과정 추가

Summary

- LU factorization can be modified to incorporate row swapping (partial pivoting).
- Solving $Ax = b$ then changes to
  - Compute P, L and U such that $PA = LU$.
  - Update $b := Pb$.
  - Solve $Lz = b$ (forward substitution).
  - Solve $Ux = z$ (backward substitution).

7.4. Inverse function

7.4.1. 역함수의 정의와 선형성

- 일반적 역함수의 정의 :

If

- $f : \mathbb{R} \to \mathbb{R}$ maps a real to a real; and
- it is a *bijection* (both one-to-one and onto)

then

- $f(x) = y$ has a unique solution for all $y \in \mathbb{R}$.
- The function that maps y to x so that $f(x) = y$ is called the inverse of f
- It is denoted by $f^{-1} : \mathbb{R} \to \mathbb{R}$.
- Importantly, $f(f^{-1}(x)) = x$ and $f^{-1}(f(x)) = x$.

→ 1대1 대응일 때, 역행렬이 존재한다.

- L(linear transformation)의 역함수가 linear transformation인가?

Let $L : \mathbb{R}^n \to \mathbb{R}^n$ be a linear transformation that is a bijection and let L^{-1} denote its inverse.

L^{-1} is a linear transformation.

Always/Sometimes/Never

Always

Let $x, y \in \mathbb{R}^n$ and $\alpha \in \mathbb{R}$.

→ linear transformation의 두 가지 조건을 만족시켜야 한다.

Let $x, y \in \mathbb{R}^n$ and $\alpha \in \mathbb{R}$.

- $L^{-1}(\alpha x) = \alpha L^{-1}(x)$: Let $u = L^{-1}(x)$. Then $x = L(u)$.

$$L^{-1}(\alpha x) = L^{-1}(\alpha L(u)) = L^{-1}(L(\alpha u)) = \alpha u = \alpha L^{-1}(x).$$

- $L^{-1}(x + y) = L^{-1}(x) + L^{-1}(y)$: Let $u = L^{-1}(x)$ and $v = L^{-1}(y)$ so that $L(u) = x$ and $L(v) = y$.

$$\begin{aligned} L^{-1}(x+y) &= L^{-1}(L(u) + L(v)) = L^{-1}(L(u+v)) \\ &= u + v = L^{-1}(x) + L^{-1}(y). \end{aligned}$$

- Theorem

Theorem

Let $L : \mathbb{R}^n \to \mathbb{R}^n$ be a linear transformation, and let A be the matrix that represents L. If there exists a matrix B such that $AB = I$, then L has an inverse, L^{-1}, and B equals the matrix that represents that linear transformation.

- Invertible : A의 역행렬이 존재하면 invertible하다고 하며, nonsingular하다고도 표현한다.

Summary

The following statements are equivalent statements about $A \in \mathbb{R}^{n \times n}$:

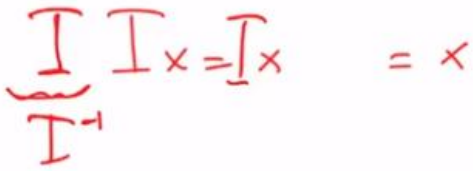
- A is nonsingular.
- A is invertible.
- A^{-1} exists.
- $AA^{-1} = A^{-1}A = I$.
- A represents a linear transformation that is a bijection.
- $Ax = b$ has a unique solution for all $b \in \mathbb{R}^n$.

7.4.2. 역함수 구하기 : 여러 예시

(1) Identity의 역함수

Inverse of the Identity matrix

If I is the identity matrix, then $I^{-1} = I$. True/False

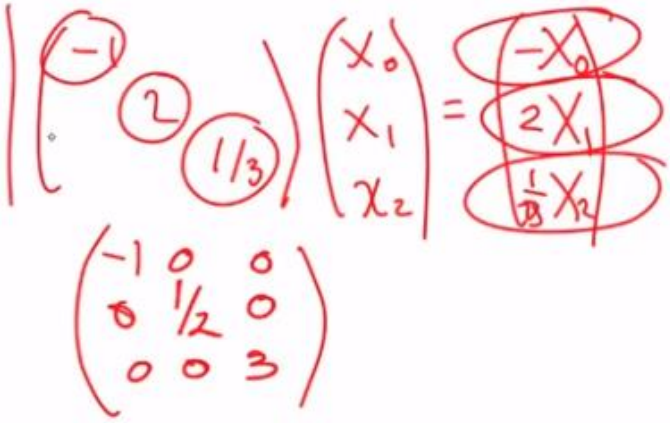


(2) 대각선만 있는 경우(Diagonal matrix) : 역수를 취하면 역행렬을 구할 수 있다.

(continued)

What effect does applying $\begin{pmatrix} -1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & \frac{1}{3} \end{pmatrix}$ to a vector have?

Answer: The first component is multiplied by -1, the second by 2 and the third by $1/3$.



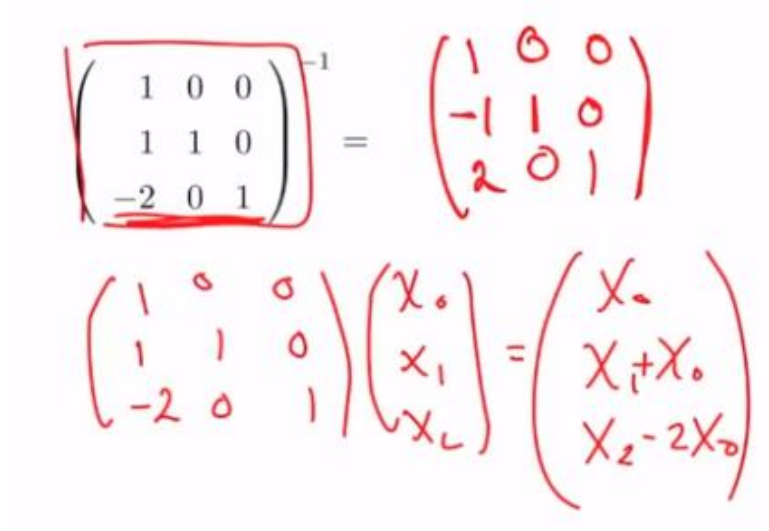
Prove the general case!

Assume $\delta_j \neq 0$ for $0 \leq j < n$.

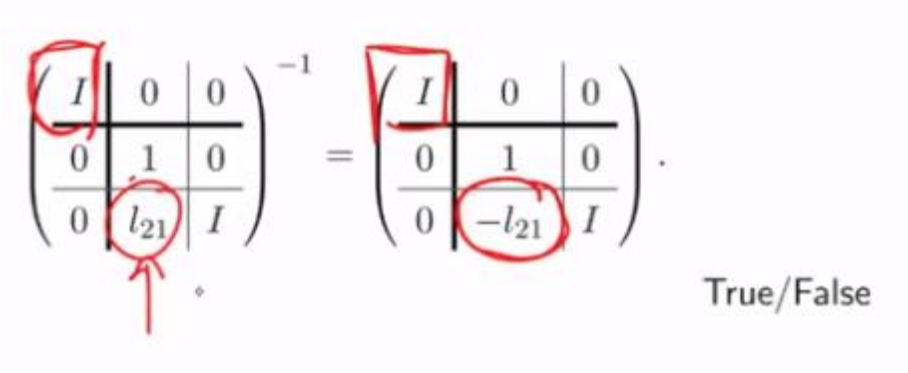
$$\begin{pmatrix} \delta_0 & 0 & \cdots & 0 \\ 0 & \delta_1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \delta_{n-1} \end{pmatrix}^{-1} = \begin{pmatrix} \frac{1}{\delta_0} & 0 & \cdots & 0 \\ 0 & \frac{1}{\delta_1} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \frac{1}{\delta_{n-1}} \end{pmatrix}.$$

Always/Sometimes/Never

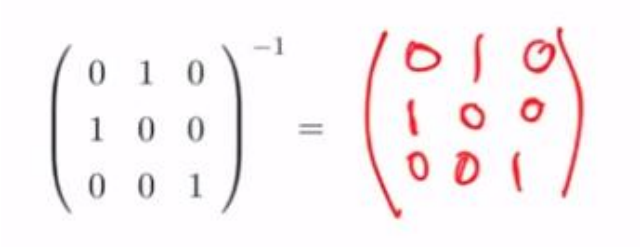
(3) Triagonal matrix : 가생이에 있는 경우 → 부호를 바꾼다.



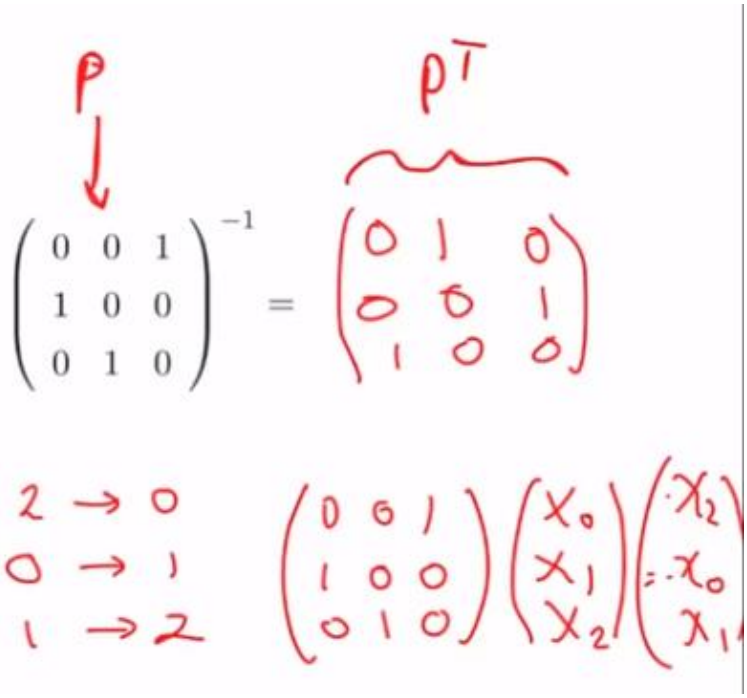
- 일반화 :



(4) permutation matrix의 역행렬은?



→ 0 → 1, 1→ 0 row를 바꾼다. ➔ 그럼 그대로인가? 아니다.



Let P be a permutation matrix. Then $P^{-1} = P$.
Always/Sometimes/Never

- 그렇다면 P의 역행렬은 transpose한 것이다.

Let P be a permutation matrix. Then $P^{-1} = P^T$.
Always/Sometimes/Never

Always

$$\underbrace{\begin{pmatrix} \chi_0 \\ \chi_1 \\ \vdots \\ \chi_{n-1} \end{pmatrix}}_{} = \underbrace{\overbrace{P^{-1}}^{B}(Px) = \begin{pmatrix} b_0 \mid b_1 \mid \cdots \mid b_{n-1} \end{pmatrix} \overbrace{\begin{pmatrix} \chi_{k_0} \\ \chi_{k_1} \\ \vdots \\ \chi_{k_{n-1}} \end{pmatrix}}^{Px}}_{}$$

→ B가 무엇인가?

$$\underbrace{\chi_0 e_0 + \chi_1 e_1 + \cdots + \chi_{n-1} e_{n-1}}_{} \qquad \chi_{k_0} b_0 + \chi_{k_1} b_1 + \cdots + \chi_{k_{n-1}} b_{n-1}$$

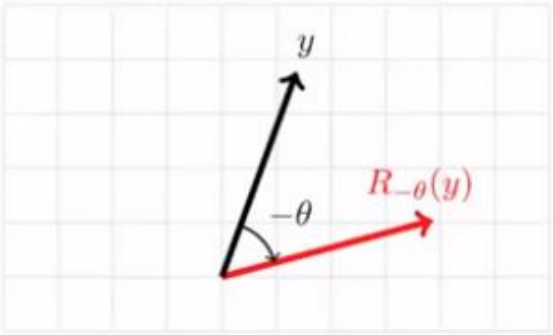
$$P^{-1} = B = \begin{pmatrix} e_{k_0} & | & e_{k_1} & | & \cdots & | & e_{k_{n-1}} \end{pmatrix} = \begin{pmatrix} e_{k_0}^T \\ e_{k_1}^T \\ \vdots \\ e_{k_{n-1}}^T \end{pmatrix}^T = P^T.$$

→ 이것으로 증명할 수 있다.

$$PP^T = \begin{pmatrix} e_{k_0}^T \\ e_{k_1}^T \\ \vdots \\ e_{k_{n-1}}^T \end{pmatrix} \begin{pmatrix} e_{k_0}^T \\ e_{k_1}^T \\ \vdots \\ e_{k_{n-1}}^T \end{pmatrix}^T = \begin{pmatrix} e_{k_0}^T \\ e_{k_1}^T \\ \vdots \\ e_{k_{n-1}}^T \end{pmatrix} \begin{pmatrix} e_{k_0} & | & e_{k_1} & | & \cdots & | & e_{k_{n-1}} \end{pmatrix}$$

$$= \begin{pmatrix} e_{k_0}^T e_{k_0} & e_{k_0}^T e_{k_1} & \cdots & e_{k_0}^T e_{k_{n-1}} \\ e_{k_1}^T e_{k_0} & e_{k_1}^T e_{k_1} & \cdots & e_{k_1}^T e_{k_{n-1}} \\ \vdots & \vdots & \ddots & \vdots \\ e_{k_{n-1}}^T e_{k_0} & e_{k_{n-1}}^T e_{k_1} & \cdots & e_{k_{n-1}}^T e_{k_{n-1}} \end{pmatrix} = \begin{pmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{pmatrix}$$

(5) 각도 회전 행렬



$$R_\theta(x): \text{ represented by } \begin{pmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{pmatrix}.$$

$$R_{-\theta}(x): \text{ represented by } \begin{pmatrix} \cos(-\theta) & -\sin(-\theta) \\ \sin(-\theta) & \cos(-\theta) \end{pmatrix}.$$

$$\begin{pmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{pmatrix}^{-1} = \begin{pmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{pmatrix}.$$

→ transpose된 행렬과 역행렬이 같다. 왜 그럴까?

7.4.3. 일반화된 역행렬 구하는 방법

- Partitioning을 해보면

General principle

$$AB = I$$

$$\underbrace{A\left(\begin{array}{c|c|c|c} b_0 & b_1 & \cdots & b_{n-1} \end{array}\right)}_{\left(\begin{array}{c|c|c|c} Ab_0 & Ab_1 & \cdots & Ab_{n-1} \end{array}\right)} = \left(\begin{array}{c|c|c|c} e_0 & e_1 & \cdots & e_{n-1} \end{array}\right)$$

$$Ab_j = e_j$$

→ 따라서, 다시 특징을 추가하자면

The following statements are equivalent statements about $A \in \mathbb{R}^{n \times n}$:

- A is nonsingular.
- A is invertible.
- A^{-1} exists.
- $AA^{-1} = A^{-1}A = I$.
- A represents a linear transformation that is a bijection.
- $Ax = b$ has a unique solution for all $b \in \mathbb{R}^n$.
- $Ax = e_j$ has a solution for all $j \in \{0, \ldots, n-1\}$.

- 풀어보자

$$\begin{pmatrix} -2 & 0 \\ 4 & 2 \end{pmatrix}^{-1} = ?$$

$$\left(\begin{array}{cc} -2 & 0 \\ 4 & 2 \end{array}\right)\left(\begin{array}{c|c} \beta_{0,0} & \beta_{0,1} \\ \beta_{1,0} & \beta_{1,1} \end{array}\right)\left(\begin{array}{c|c} 1 & 0 \\ 0 & 1 \end{array}\right)$$

$$\begin{pmatrix} -2 & 0 \\ 4 & 2 \end{pmatrix}\begin{pmatrix} \beta_{0,0} \\ \beta_{1,0} \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \quad \text{yields} \quad \begin{pmatrix} \beta_{0,0} \\ \beta_{1,0} \end{pmatrix} = \begin{pmatrix} -\frac{1}{2} \\ 1 \end{pmatrix}$$

$$\begin{pmatrix} -2 & 0 \\ 4 & 2 \end{pmatrix}\begin{pmatrix} \beta_{0,1} \\ \beta_{1,1} \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \quad \text{yields} \quad \begin{pmatrix} \beta_{0,1} \\ \beta_{1,1} \end{pmatrix} = \begin{pmatrix} 0 \\ \frac{1}{2} \end{pmatrix}$$

- 일반화하기

$$L = \left(\begin{array}{c|c} L_{00} & 0 \\ \hline l_{10}^T & \lambda_{11} \end{array}\right)$$

$$L^{-1} = \left(\begin{array}{c|c} L_{00}^{-1} & 0 \\ \hline -\frac{1}{\lambda_{11}} l_{10}^T L_{00}^{-1} & \frac{1}{\lambda_{11}} \end{array}\right)$$

$$\left(\begin{array}{c|c} L_{00} & 0 \\ \hline l_{10}^T & \lambda_{11} \end{array}\right)\left(\begin{array}{c|c} L_{00}^{-1} & 0 \\ \hline -\frac{l_{10}^T L_{00}^{-1}}{\lambda_{11}} & \frac{1}{\lambda_{11}} \end{array}\right)$$

$$= \left(\begin{array}{c|c} L_{00}L_{00}^{-1} & L_{00}0 + 0\frac{1}{\lambda_{11}} \\ \hline -l_{10}^T L_{00}^{-1} + \lambda_{11}\frac{l_{10}^T L_{00}^{-1}}{\lambda_{11}} & 0 \times 0 + \lambda_{11}\frac{1}{\lambda_{11}} \end{array}\right)$$

-

- 2x2 matrix의 역행렬 구하기 일반화

If $\alpha_{0,0}\alpha_{1,1} - \alpha_{1,0}\alpha_{0,1} \neq 0$

$$\left(\begin{array}{c|c} \alpha_{0,0} & \alpha_{0,1} \\ \hline \alpha_{1,0} & \alpha_{1,1} \end{array}\right)^{-1} = \frac{1}{\alpha_{0,0}\alpha_{1,1} - \alpha_{1,0}\alpha_{0,1}} \left(\begin{array}{c|c} \alpha_{1,1} & -\alpha_{0,1} \\ \hline -\alpha_{1,0} & \alpha_{0,0} \end{array}\right)$$

Inverses of special matrices

Type	A	A^{-1}
Identity matrix	$I = \begin{pmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \end{pmatrix}$	$I = \begin{pmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{pmatrix}$
Diagonal matrix	$D = \begin{pmatrix} \delta_{0,0} & 0 & \cdots & 0 \\ 0 & \delta_{1,1} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \delta_{n-1,n-1} \end{pmatrix}$	$D^{-1} = \begin{pmatrix} \delta_{0,0}^{-1} & 0 & \cdots & 0 \\ 0 & \delta_{1,1}^{-1} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \delta_{n-1,n-1}^{-1} \end{pmatrix}$
Gauss transform	$\widetilde{L} = \left(\begin{array}{c\|c\|c} I & 0 & 0 \\ \hline 0 & 1 & 0 \\ \hline 0 & l_{21} & I \end{array}\right)$	$\widetilde{L}^{-1} = \left(\begin{array}{c\|c\|c} I & 0 & 0 \\ \hline 0 & 1 & 0 \\ \hline 0 & -l_{21} & I \end{array}\right)$.
Permutation matrix	P	P^T
2D Rotation	$R = \begin{pmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{pmatrix}$	$R^{-1} = \begin{pmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{pmatrix} = R^T$
2D Reflection	A	A
Lower triangular matrix	$L = \left(\begin{array}{c\|c} L_{00} & 0 \\ \hline l_{10}^T & \lambda_{11} \end{array}\right)$	$L^{-1} = \left(\begin{array}{c\|c} L_{00}^{-1} & 0 \\ \hline -\frac{1}{\lambda_{11}} l_{10}^T L_{00}^{-1} & \frac{1}{\lambda_{11}} \end{array}\right)$
Upper triangular matrix	$U = \left(\begin{array}{c\|c} U_{00} & u_{01} \\ \hline 0 & v_{11} \end{array}\right)$	$U^{-1} = \left(\begin{array}{c\|c} U_{00}^{-1} & -U_{00}^{-1} u_{01}/v_{11} \\ \hline 0 & \frac{1}{v_{11}} \end{array}\right)$
General 2×2 matrix	$\begin{pmatrix} \alpha_{0,0} & \alpha_{0,1} \\ \alpha_{1,0} & \alpha_{1,1} \end{pmatrix}$	$\frac{1}{\alpha_{0,0}\alpha_{1,1} - \alpha_{1,0}\alpha_{0,1}} \begin{pmatrix} \alpha_{1,1} & -\alpha_{0,1} \\ -\alpha_{1,0} & \alpha_{0,0} \end{pmatrix}$

7.4.4. 역행렬의 성질(Properties)

Properties of the inverse

Assume A, B, and C are square matrices that are nonsingular. Then

- $(\alpha B)^{-1} = \frac{1}{\alpha} B^{-1}$.
- $(AB)^{-1} = B^{-1}A^{-1}$.
- $(ABC)^{-1} = C^{-1}B^{-1}A^{-1}$.
- $(A^T)^{-1} = (A^{-1})^T$.
- $(A^{-1})^{-1} = A$.

The following statements are equivalent statements about $A \in \mathbb{R}^{n \times n}$:

- A is nonsingular.
- A is invertible.
- A^{-1} exists.
- $AA^{-1} = A^{-1}A = I$.
- A represents a linear transformation that is a bijection.
- $Ax = b$ has a unique solution for all $b \in \mathbb{R}^n$.
- $Ax = 0$ implies that $x = 0$.
- LU factorization with row (partial) pivoting yields an upper triangular matrix with no zeroes on its diagonal.
- $Ax = e_j$ has a solution for all $j \in \{0, \ldots, n-1\}$.
- The determinant of A is nonzero: $\det(A) \neq 0$.

9. Vector spaces

<2단원. 선형대수학을 배우는 이유>

2.2. linear algebra의 중요성

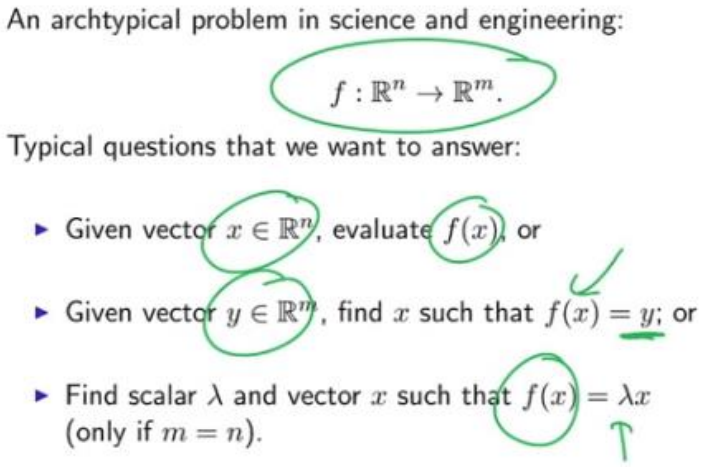
F는 vector를 input으로 vector가 output으로 나오는 함수다.

(1) f 를 알고 y를 알 때, 해 x를 구하고 싶다.

(2) f를 아는데, 이를 스칼라랑 x의 곱으로 표현하고 싶다!!! ➔ Eigen value problem

→ f가 linear transformation일 때, 이런 문제가 쉽게 풀린다. (linear가 아니면 local 구간을 정해서 선형으로 근사를 한다.)

➔ linear transformations are a subset of vector functions for which these problems are simpler to solve.



< 큰 그림 요약 >

Now, we can see that when executing Gaussian elimination (LU factorization) with Ax=b where A is a square matrix, one of three things can happen:

(1) 가능성 1. LU factorization이 성공적으로 되고 U 대각선에 0이 없는 경우 ➔ 유일한 해

The process completes with no zeroes on the diagonal of the resulting matrix U. Then A=LU and Ax=b has a unique solution, which can be found by solving Lz=b followed by Ux=z.

(2) 가능성 2. Row를 바꿔야하는 경우에 ➔ Lz=Pb를 한다(**Permutation**). ➔ 유일한 해를 가진다.

The process requires row exchanges, completing with no zeroes on the diagonal of the resulting matrix U. Then PA=LU and Ax=b has a unique solution, which can be found by solving Lz=Pb followed by Ux=z.

(3) 가능성 3. Pivoting을 하더라도 대각선에 0이 생길 수 밖에 없는 경우이다. ➔ 해가 없거나(불능이거나) 부정이다.

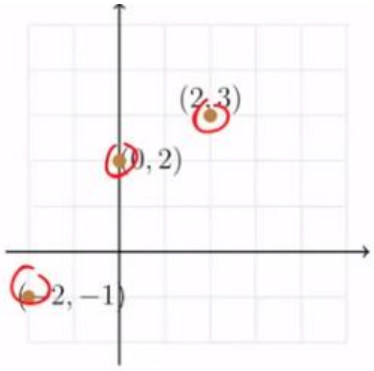
The process requires row exchanges, but at some point no row can be found that puts a nonzero on the diagonal, at which point the process fails (unless the zero appears as the last element on the diagonal, in which case it completes, but leaves a zero on the diagonal).

9.1. Coefficient를 구하는 과정!

$$p(\chi) = \gamma_0 + \gamma_1\chi + \gamma_2\chi^2.$$

We know that

- $p(-2) = -1$,
- $p(0) = 2$
- $p(2) = 3$.



$$\begin{array}{rcl} p(-2) & = & \gamma_0 + \gamma_1(-2) + \gamma_2(-2)^2 = -1 \\ p(0) & = & \gamma_0 + \gamma_1(0) + \gamma_2(0)^2 = 2 \\ p(2) & = & \gamma_0 + \gamma_1(2) + \gamma_2(2)^2 = 3 \end{array}$$

(1) polynominal equation의 계수(coefficient)를 일반적으로 구하는 법

In matrix notation

$$\begin{pmatrix} 1 & -2 & 4 \\ 1 & 0 & 0 \\ 1 & 2 & 4 \end{pmatrix} \begin{pmatrix} \gamma_0 \\ \gamma_1 \\ \gamma_2 \end{pmatrix} = \begin{pmatrix} -1 \\ 2 \\ 3 \end{pmatrix}.$$

Yielding

$$\begin{pmatrix} \gamma_0 \\ \gamma_1 \\ \gamma_2 \end{pmatrix} = \begin{pmatrix} 2 \\ 1 \\ -\frac{1}{4} \end{pmatrix}$$

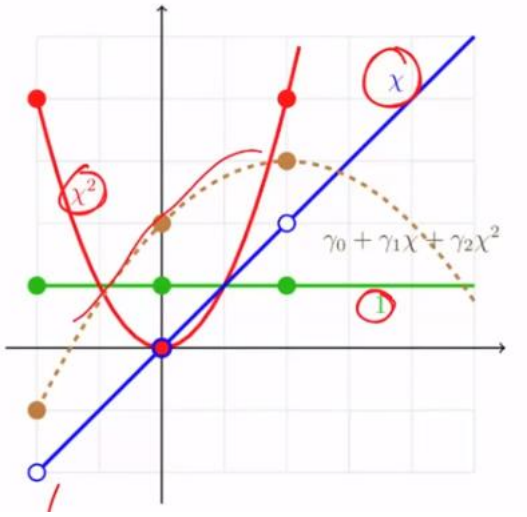
so that

$$p(\chi) = 2 + \chi - \frac{1}{4}\chi^2.$$

(2) 이 문제를 다른 관점에서 보는 방법

저 방정식을 $P_0(x)=1$, $P_1(x)=x$, $P_2(x)=x^2$의 linear combination으로 보게 문제를 재정의해보자.

Different view of same problem



→ parent function ($P_0(x)=1$, $P_1(x)=x$, $P_2(x)=x^2$)을 어떻게 조합해서 갈색의 function을 만들 수 있을까?

$$p(x) = \gamma_0 x^0 + \gamma_1 x^1 + \gamma_2 x^2$$

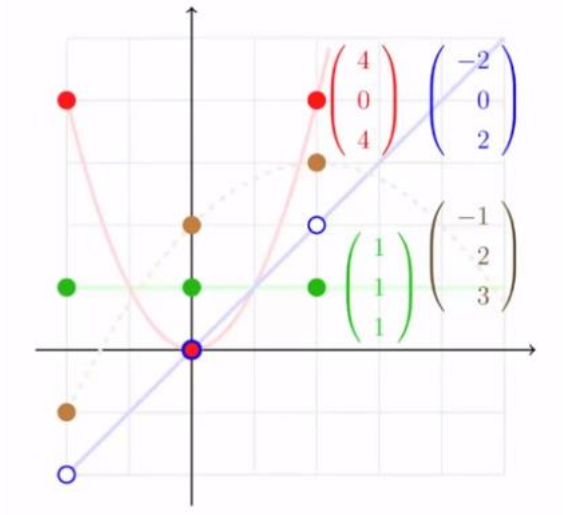
$$\begin{matrix} p(-2) \\ p(0) \\ p(2) \end{matrix} = \begin{matrix} \gamma_0(-2)^0 \\ \gamma_0(0)^0 \\ \gamma_0(2)^0 \end{matrix} + \begin{matrix} \gamma_1(-2)^1 \\ \gamma_1(0)^1 \\ \gamma_1(2)^1 \end{matrix} + \begin{matrix} \gamma_2(-2)^2 \\ \gamma_2(0)^2 \\ \gamma_2(2)^2 \end{matrix}$$

$$\begin{pmatrix} p(-2) \\ p(0) \\ p(2) \end{pmatrix} = \gamma_0 \begin{pmatrix} (-2)^0 \\ (0)^0 \\ (2)^0 \end{pmatrix} + \gamma_1 \begin{pmatrix} (-2)^1 \\ (0)^1 \\ (2)^1 \end{pmatrix} + \gamma_2 \begin{pmatrix} (-2)^2 \\ (0)^2 \\ (2)^2 \end{pmatrix}$$

$$\begin{pmatrix} -1 \\ 2 \\ 3 \end{pmatrix} = \gamma_0 \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} + \gamma_1 \begin{pmatrix} -2 \\ 0 \\ 2 \end{pmatrix} + \gamma_2 \begin{pmatrix} 4 \\ 0 \\ 4 \end{pmatrix}.$$

→ (-2, 0, 2)를 조합해서 넣어봤을 때, 만든 식이다.

→ 세 개의 벡타로 갈색의 벡타를 표현할 linear transformation은 없을까? 각각의 벡타는 각 parent function을 represent하는 것인데, 어떤 linear combination을 하면 discretized target function을 대변하는 vector가 될까?



$$\gamma_0 \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} + \gamma_1 \begin{pmatrix} -2 \\ 0 \\ 2 \end{pmatrix} + \gamma_2 \begin{pmatrix} 4 \\ 0 \\ 4 \end{pmatrix} = \begin{pmatrix} -1 \\ 2 \\ 3 \end{pmatrix}$$

$$\begin{pmatrix} 1 & -2 & 4 \\ 1 & 0 & 0 \\ 1 & 2 & 4 \end{pmatrix} \begin{pmatrix} \gamma_0 \\ \gamma_1 \\ \gamma_2 \end{pmatrix} = \begin{pmatrix} -1 \\ 2 \\ 3 \end{pmatrix}$$

→ 이렇게 간단한 matrix-vector multiplication 문제로 환원된다.

종합 : 정리.

Two views of polynomial interpolation

- **View 1:** Satisfy constraints imposed by the fact that the target function must go through a given set of points:

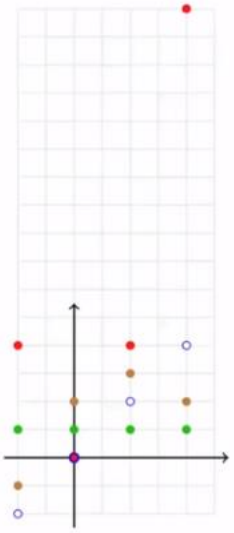
$$\begin{aligned} p(-2) &= \gamma_0 + \gamma_1(-2) + \gamma_2(-2)^2 &= -1 \\ p(0) &= \gamma_0 + \gamma_1\,(0) + \gamma_2\,(0)^2 &= 2 \\ p(2) &= \gamma_0 + \gamma_1\,(2) + \gamma_2\,(2)^2 &= 3 \end{aligned}$$

- **View 2:** Find the linear combination of vectors that represent the parent functions to yield the vector that represents the target function.

$$\gamma_0 \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} + \gamma_1 \begin{pmatrix} -2 \\ 0 \\ 2 \end{pmatrix} + \gamma_2 \begin{pmatrix} 4 \\ 0 \\ 4 \end{pmatrix} = \begin{pmatrix} -1 \\ 2 \\ 3 \end{pmatrix}$$

9.2. 해가 반드시 존재하지는 않는다. 해가 없는 경우.

만약에 4개의 점이 있다면?



$$p(x) = \gamma_0 + \gamma_1 x + \gamma_2 x^2$$

where

- $p(-2) = -1$
- $p(0) = 2$
- $p(2) = 3$
- $p(4) = 2$

$$\begin{pmatrix} 1 & -2 & 4 \\ 1 & 0 & 0 \\ 1 & 2 & 4 \\ 1 & 4 & 16 \end{pmatrix} \begin{pmatrix} \gamma_0 \\ \gamma_1 \\ \gamma_2 \end{pmatrix} = \begin{pmatrix} -1 \\ 2 \\ 3 \\ 2 \end{pmatrix}$$

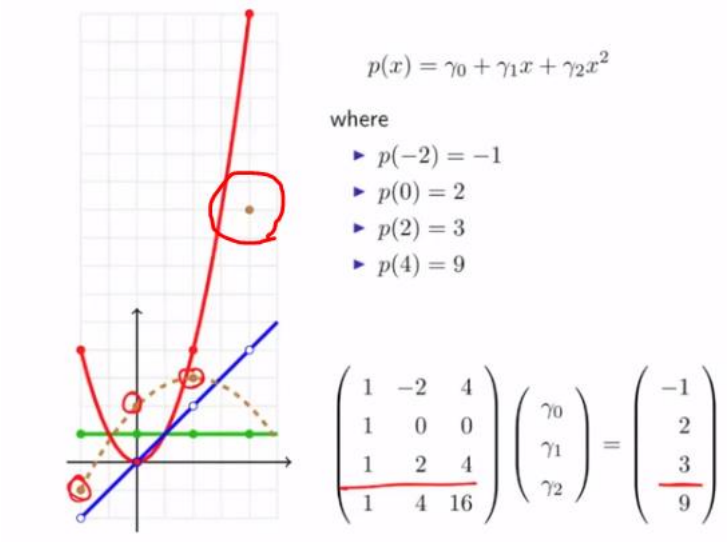
is equivalent to

$$\begin{pmatrix} 1 & -2 & 4 \\ 1 & 0 & 0 \\ 1 & 2 & 4 \end{pmatrix} \begin{pmatrix} \gamma_0 \\ \gamma_1 \\ \gamma_2 \end{pmatrix} = \begin{pmatrix} -1 \\ 2 \\ 3 \end{pmatrix}$$

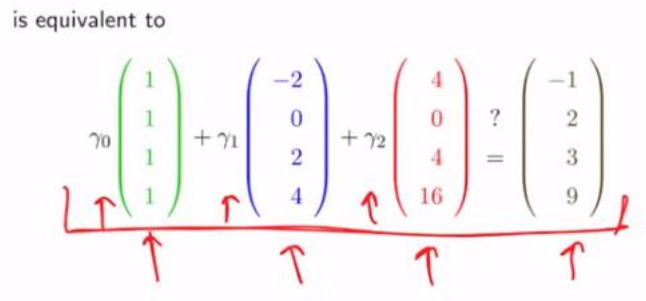
$$\begin{pmatrix} 1 & 4 & 16 \end{pmatrix} \begin{pmatrix} \gamma_0 \\ \gamma_1 \\ \gamma_2 \end{pmatrix} = \begin{pmatrix} 2 \end{pmatrix}.$$

→ slicing과 dicing을 하면 된다.

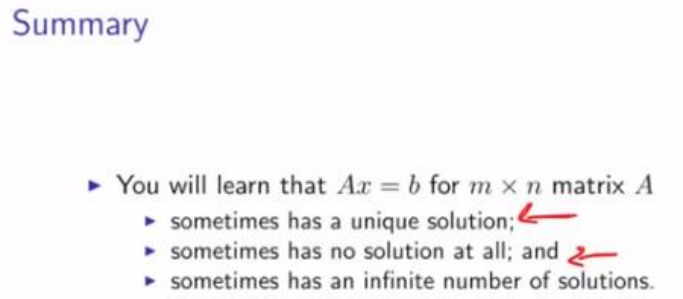
- 해가 없는 경우가 있다.



→ 만약에 빨간 동그라미 안의 갈색 점을 추가하면 어떻게 될까? : 해가 없게 된다. 다른 각도로 바라보면,

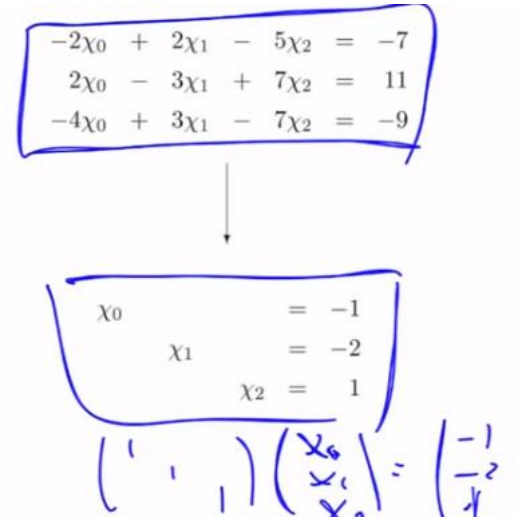


→ linear combination을 만족할 계수가 있을까? 어떤 조건 하에서 이 계수가 존재할 것인가? 그것은 A함수와 right hand side의 b가 결정한다.

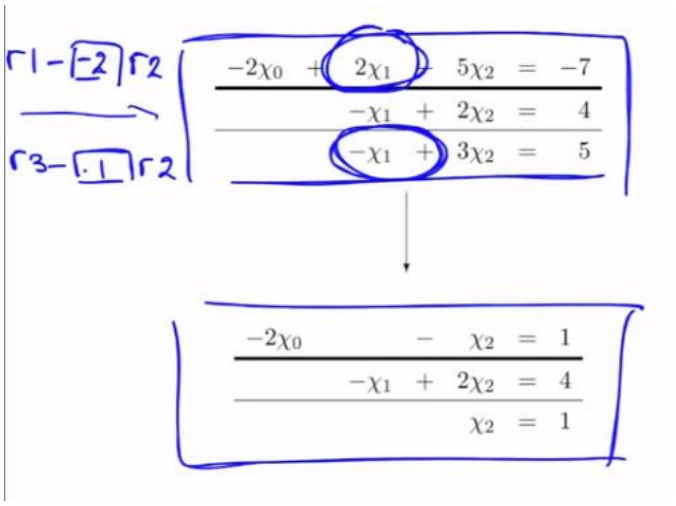


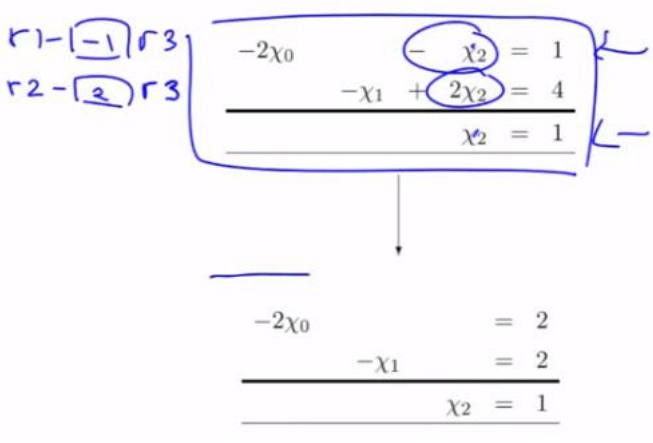
<8단원 복습>

- gauss-jordan elimination이란 : diagonal matrix만 남기고 없애는 과정까지 진행한 것이다. 이것도 마찬가지로



- 과정





9.3. 해가 여러 개인 경우

- Gauss-Jordan elimination의 적용

Apply Gauss-Jordan elimination

$$\left(\begin{array}{ccc|c} 2 & 2 & -2 & 0 \\ -2 & -3 & 4 & 3 \\ 4 & 3 & -2 & 4 \end{array}\right) \longrightarrow \left(\begin{array}{ccc|c} 2 & 2 & -2 & 0 \\ 0 & -1 & 2 & 3 \\ 0 & -1 & 2 & 4 \end{array}\right) \longrightarrow$$

$$\left(\begin{array}{ccc|c} 2 & 0 & 2 & 6 \\ 0 & -1 & 2 & 3 \\ 0 & 0 & 0 & 1 \end{array}\right) \longrightarrow$$

$$\begin{array}{rcl} \chi_0 + \chi_2 & = & 3 \\ -\chi_1 - 2\chi_2 & = & -3 \\ 0 & = & 1 \end{array}$$

→ 0 = 1이 나오기 때문에 이것은 해가 없는 것이다.

- 해가 여러 개 일 때, general solution을 찾는 방법.

Consider $Ax = b$. Assume that we have

- A specific (particular) solution x_s so that $Ax_s = b$.
- Solution x_n with $Ax_n = 0$.

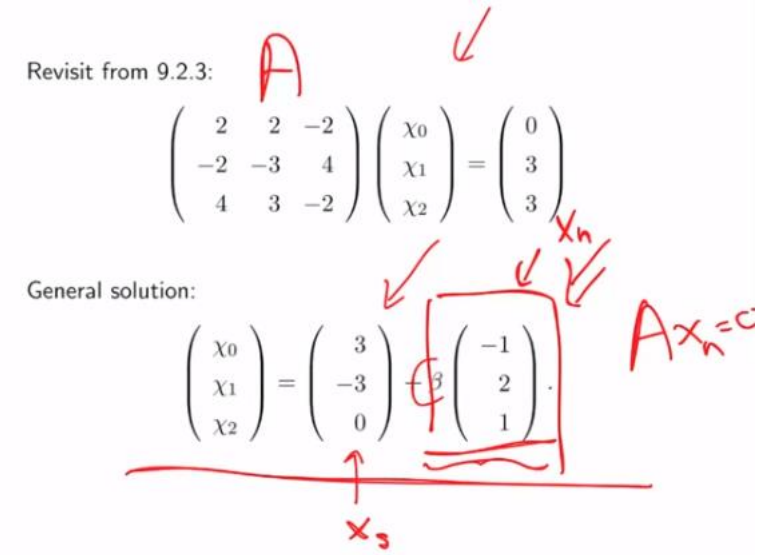
Then

$$\begin{array}{ll} A(x_s + x_n) & \\ = & < \text{Distribute } A > \\ Ax_s + Ax_n & \\ = & < Ax_s = b \text{ and } Ax_n = 0 > \\ b + 0 & \\ = & < \text{algebra} > \\ b & \end{array}$$

Under the same assumptions

$$\begin{array}{ll} A(x_s + \beta x_n) & = b \\ = & < \text{Distribute } A > \\ Ax_s + A(\beta x_n) & \\ = & < Ax \text{ is a linear transformation} > \\ Ax_s + \beta Ax_n & \\ = & < Ax_s = b \text{ and } Ax_n = 0 > \\ b + 0 & \\ = & < \text{algebra} > \\ b & \end{array}$$

- How to find solutions



- Systematic하게 구하는 법 : 8장에서 배운 듯이 두 식을 동시에 풀 수 있는 방법을 사용하자.

Summary

Given $Ax = b$

- Find a specific solution, x_s with $Ax_s = b$.
- Unique solution? Done!
- Find vector(s) x_n such that $Ax_n = 0$ and use it (these) to specify the general solution.

We will make this procedure more precise later this week.

(1) 단계1 : 행렬식을 gauss-jordan elimination과 유사하게 diagonal하게 만든다.

$$\underbrace{\left(\begin{array}{ccc|c} 2 & 2 & -2 & 0 \\ -2 & -3 & 4 & 3 \\ 4 & 3 & -2 & 3 \end{array}\right) \quad \left(\begin{array}{ccc|c} 2 & 2 & -2 & 0 \\ -2 & -3 & 4 & 0 \\ 4 & 3 & -2 & 0 \end{array}\right)}_{\left(\begin{array}{ccc|cc} 2 & 2 & -2 & 0 & 0 \\ -2 & -3 & 4 & 3 & 0 \\ 4 & 3 & -2 & 3 & 0 \end{array}\right)}$$

$$\left(\begin{array}{ccc|cc} 2 & 2 & -2 & 0 & 0 \\ -2 & -3 & 4 & 3 & 0 \\ 4 & 3 & -2 & 3 & 0 \end{array}\right) \longrightarrow \left(\begin{array}{ccc|cc} \boxed{2} & 2 & -2 & 0 & 0 \\ 0 & -1 & 2 & 3 & 0 \\ 0 & -1 & 2 & 3 & 0 \end{array}\right) \longrightarrow$$

$$\left(\begin{array}{ccc|cc} \boxed{2} & 2 & -2 & 0 & 0 \\ 0 & \boxed{-1} & 2 & 3 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{array}\right) \longrightarrow \left(\begin{array}{ccc|cc} \boxed{2} & 0 & 2 & 6 & 0 \\ 0 & \boxed{-1} & 2 & 3 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{array}\right) \longrightarrow$$

$$\left(\begin{array}{ccc|cc} \boxed{1} & 0 & 1 & 3 & 0 \\ 0 & \boxed{1} & -2 & -3 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{array}\right) \longrightarrow$$

$$\begin{array}{rcrcl} \chi_0 & & + \chi_2 & = & 3 \\ & \chi_1 & - 2\chi_2 & = & -3 \\ & & 0 & = & 0 \end{array} \quad \text{and} \quad \begin{array}{rcrcl} \chi_0 & & + \chi_2 & = & 0 \\ & \chi_1 & - 2\chi_2 & = & 0 \\ & & 0 & = & 0 \end{array}$$

→ 사실 0 0 0 쪽은 gauss transformation을 해도 0 0 0이기 때문에 신경을 크게 안 써도 된다.

(2) 단계2 : finding a specific solution (Ax=b)

→ free variable을 찾고, 편한 값을 대입해서 x0, x1을 구한다.

pivot을 했던 x2는 free variable이다. x0, x1은 dependent variable이다.(free variable에 따라서 값이 바뀌기 때문이다). x2 = 0라고 하자. 그러면 x0=3. 그런데 pivot이 뭐지?

Find a specific (particular) solution

$$\begin{array}{rcrcl} \boxed{\chi_0} & & + \chi_2 & = & 3 \\ & \boxed{\chi_1} & - 2\chi_2 & = & -3 \\ & & 0 & = & 0 \end{array}$$

free variable

Find a specific (particular) solution

$$\begin{array}{rcrcl} \chi_0 & & + \chi_2 & = & 3 \\ & \chi_1 & - 2\chi_2 & = & -3 \\ & & 0 & = & 0 \end{array}$$

Choose the free variable $\chi_2 = 0$.

$$\begin{array}{rcrcl} \chi_0 & & + 0 & = & 3 \\ & \chi_1 & - 2(0) & = & -3 \\ & & 0 & = & 0 \end{array}$$

Solve:

$$x_s = \begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix} = \begin{pmatrix} 3 \\ -3 \\ 0 \end{pmatrix}.$$

(3) 단계3 : Ax=0을 만족하는 벡터 찾기. find a solution that maps to zero

Find a vector that maps to zero

$$\begin{aligned} \chi_0 \quad + \quad \chi_2 &= 0 \\ \chi_1 - 2\chi_2 &= 0 \\ 0 &= 0 \end{aligned}$$

Choose the free variable $\chi_2 = 1$.

$$\begin{aligned} \chi_0 \quad + \quad (1) &= 0 \\ \chi_1 - 2(1) &= 0 \\ 0 &= 0 \end{aligned}$$

Solve:

$$x_s = \begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix} = \begin{pmatrix} -1 \\ 2 \\ 1 \end{pmatrix}.$$

(4) 마무리 : 그 두 개의 vector를 더한 형태가 답이다. 2개의 벡터만으로도 무한개의 solution을 표현할 수 있다.

General solution

$$x = x_s + \beta x_n = \begin{pmatrix} 3 \\ -3 \\ 0 \end{pmatrix} + \beta \begin{pmatrix} -1 \\ 2 \\ 1 \end{pmatrix}.$$

General solution

9.4. 집합과 벡터 개념

- R^n의 개념 : huge set of vectors

The set of all vectors of size n whose components are real valued is denoted by $\mathbb{R}^n$.

- Span의 개념 : Span of a set of vectors

  - all possible vectors that can be generated by a set of vectors. (linear combination)

Let $\{v_0, v_1, \cdots, v_{n-1}\} \subset \mathbb{R}^m$.

$\text{Span}\{v_0, v_1, \cdots, v_{n-1}\}$: the set of all vectors that are a linear combination of the given set of vectors.

Example

$$\left\{ \alpha_0 \begin{pmatrix} 1 \\ 0 \end{pmatrix} + \alpha_1 \begin{pmatrix} 0 \\ 1 \end{pmatrix} \middle| \alpha_0, \alpha_1 \in \mathbb{R} \right\}$$

All of $\mathbb{R}^2$ (an uncountable infinite set) described with two vectors.

9.4.1. 벡터 공간과 조건

- 배우는 이유 : why few vectors can describe an infinite number of vectors in the vector space.
- 벡터 공간의 조건 :

(1) 0벡터 포함 [0 0 0 0 0 0 .... 0]

(2) 덧셈연산에 닫혀있어야 한다.(closure under addition) a in V, b in V → a+b in V

(3) scaling 연산에 닫혀있어야 한다. (closure under scalar multiplication) a in V, c in R → c*a in V

For our purposes...

A vector space is a subset, S, of $\mathbb{R}^n$ with the following properties:

- $0 \in S$ (the zero vector of size n is in the set S); and
- If $v, w \in S$ then $(v + w) \in S$; and
- If $\alpha \in \mathbb{R}$ and $v \in S$ then $\alpha v \in S$.

"S is closed under addition and scalar multiplication."

All the results that we will encounter for such vector spaces carry over to the case where the components of vectors are complex valued.

9.4.2. Subspaces of R^n : R^n의 부분집합 중에서 벡터공간인 것

(1) subspace의 정의와 조건 :

Homework 9.4.2.1: Is this a subspace?

The plane of vectors $x = \begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix}$ such that $\chi_0 = 0$:

$$\left\{ x \in \mathbb{R}^3 \middle| x = \begin{pmatrix} 0 \\ \chi_1 \\ \chi_2 \end{pmatrix} \right\}$$

Yes.

- $x + y$ is in the set: If x and y are in the set, then $x = \begin{pmatrix} 0 \\ \chi_1 \\ \chi_2 \end{pmatrix}$ and $y = \begin{pmatrix} 0 \\ \psi_1 \\ \psi_2 \end{pmatrix}$. But then $x + y = \begin{pmatrix} 0 \\ \chi_1 + \psi_1 \\ \chi_2 + \psi_2 \end{pmatrix}$ is in the set.
- αx is in the set: If x is in the set and $\alpha \in \mathbb{R}$, then $\alpha x = \begin{pmatrix} 0 \\ \alpha\chi_1 \\ \alpha\chi_2 \end{pmatrix}$ is in the set.

- 좋은 예시 :

Homework 9.4.2.1: Is this a subspace?

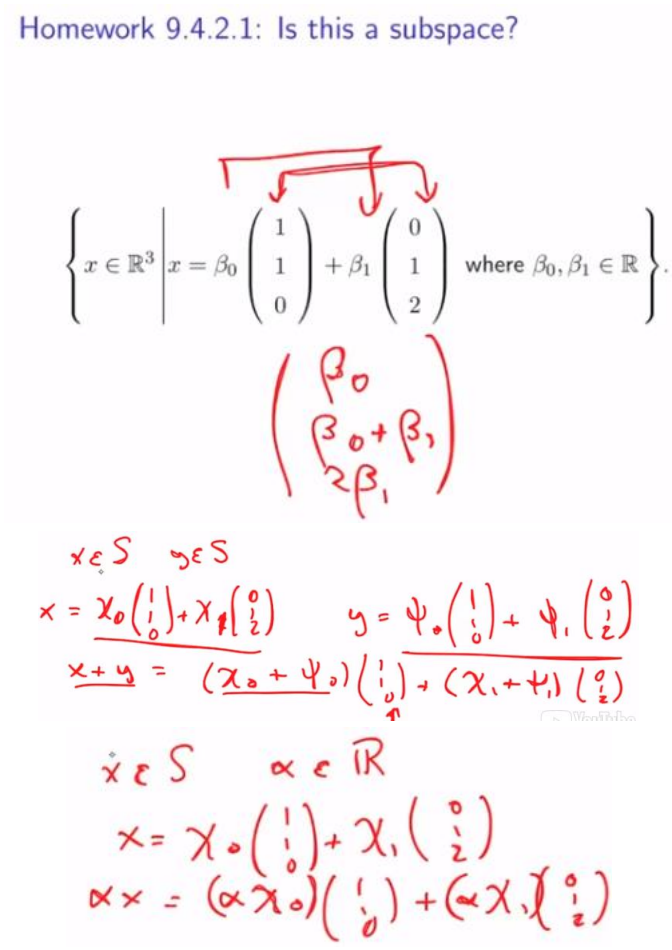
$$\left\{ x \in \mathbb{R}^3 \middle| x = \begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix} \wedge \chi_0\chi_1 = 0 \right\}$$

No

$x + y$ is not in the set if $x = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$ and $y = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}$.

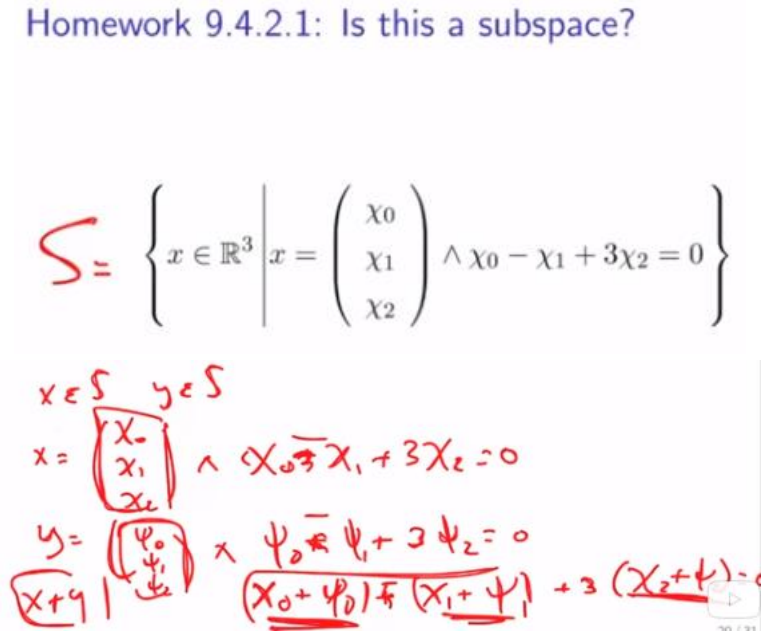
→ 와 이거 나 답 틀렸다. 더해서 반례를 만들 수 있다.

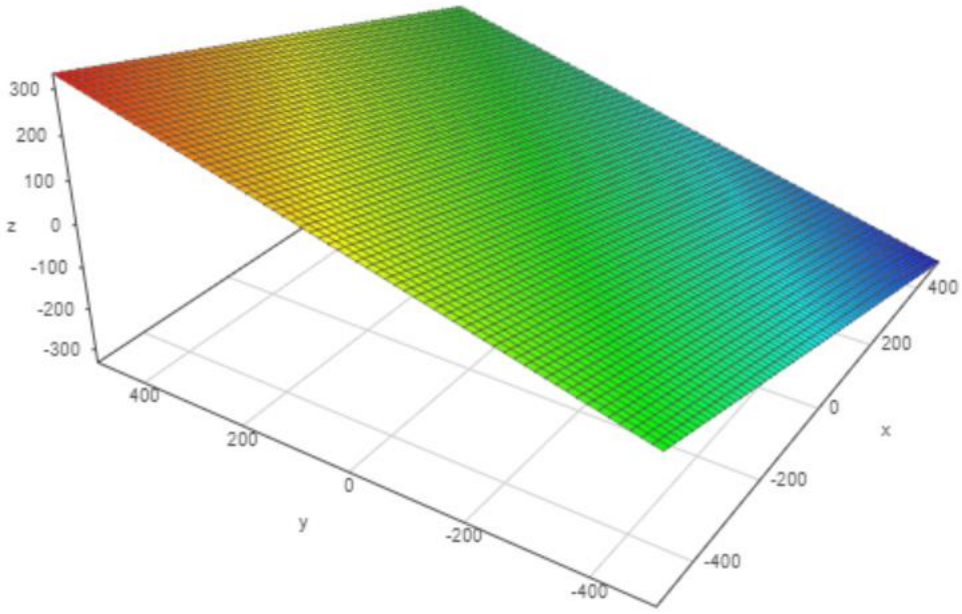
- linear combination of two vector로 만든 것은 subspace일까?



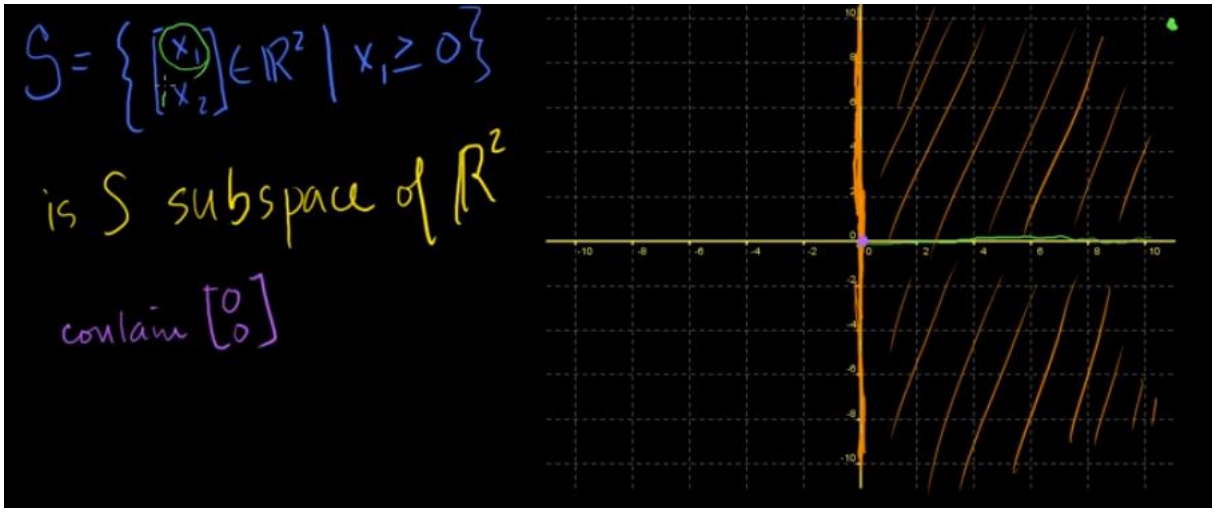
→ subspace가 맞다. (1) 영벡타있고 (2) 덧셈 (3) 스칼라곱에도 닫혀있다.

- 예시2 :





- 예시3 :



- 그렇다면, 3개의 벡터의 linear combination (span)도 subspace일까?
- 그렇다면, n개의 벡터의 linear combination (span)도 subspace일까?

(2) subspace의 표현 : 역으로 subspace를 어떻게 표현할 수 있을까?

The purpose of the game:

- Given a subspace S,
- Find the spanning set for S with fewest vectors.

- 이를 위해서 linear independence의 개념을 도입한다.
- linearly independent한 벡타들이 subspace S를 span하면 그것이 basis for Subspace이 된다.
- basis의 개념

Definition

Let S be a subspace of $\mathbb{R}^m$. Then the set

$$\{v_0, v_1, \cdots, v_{n-1}\} \subset \mathbb{R}^m$$

is said to be a *basis* for S if

1. $\{v_0, v_1, \cdots, v_{n-1}\}$ are linearly independent; and
2. $\text{Span}(\{v_0, v_1, \cdots, v_{n-1}\}) = S$.

- 예시 : R^n의 basis

Homework

The vectors $\{e_0, e_1, \ldots, e_{n-1}\} \subset \mathbb{R}^n$ are a basis for $\mathbb{R}^n$.

True/False

True

- Clearly, $\text{Span}(e_0, e_1, \ldots, e_{n-1}) = \mathbb{R}^n$.
- Now, the identity $I = \left(\begin{array}{c|c|c|c} e_0 & e_1 & \cdots & e_{n-1} \end{array} \right)$.
- $Ix = 0$ only has the solution $x = 0$.
- Hence the columns of I are linearly independent which means the vectors $\{e_0, e_1, \ldots, e_{n-1}\}$ are linearly independent.

- 응용 : invertible한 경우에는 basis가 된다.

Let $\{a_0, \ldots, a_{n-1}\} \subset \mathbb{R}^n$ and let $A = \left(\begin{array}{c|c|c|c} a_0 & a_1 & \cdots & a_{n-1} \end{array} \right)$ be invertible.

Then $\{a_0, \ldots, a_{n-1}\} \subset \mathbb{R}^n$ form a basis for $\mathbb{R}^n$.

→ 조건 (1) invertible하면 linearly independent하다. Ax = 0일 때, null space가 0밖에 없기 때문이다.

(2) Invertible하다는 것은 Ax = b에서 해가 있다는 것이다. 따라서 어떤 b의 subspace든 간에 Ax로 linear combination을 통해서 표현할 수 있다.

- 조건 (1) 부가 : 질문!!! 왜???

Definitions:

- A is invertible if there exists a matrix A^{-1} such that $AA^{-1} = A^{-1}A = I$
- The vectors $v_1, \ldots, v_n$ are linearly independent if the only solution to $x_1v_1 + \cdots + x_nv_n = 0$ (with $x_i \in \mathbb{R}$) is $x_1 = \cdots = x_n = 0$.

Textbook Proof:

> **Fact:** With $v_1, \ldots, v_n$ referring to the columns of A, the equation $x_1v_1 + \cdots + x_nv_n = 0$ can be rewritten as $Ax = 0$. (This is true by definition of matrix multiplication)

Now, suppose that A is invertible. We want to show that the only solution to $Ax = 0$ is $x = 0$ (and by the above fact, we'll have proven the statement).

Multiplying both sides by A^{-1} gives us

$$Ax = 0 \implies A^{-1}Ax = A^{-1}0 \implies x = 0$$

So, we may indeed state that the only x with $Ax = 0$ is the vector $x = 0$.

- m차원의 벡터 subspace에는 최대 m개의 linearly independent vector가 있을 것이다.

Lemma

Let $S \subset \mathbb{R}^m$. Then S contains at most m linearly independent vectors.

Proof

Proof by contradiction:
Assume that S contains more than m linearly independent vectors.
We will show that this leads to a contradiction.

- Since S contains more than m linearly independent vectors, it contains at least $m+1$ linearly independent vectors.
- Let us label $m+1$ such vectors $v_0, v_1, \ldots, v_{m-1}, v_m$.
- Let $V = \left(\begin{array}{c|c|c|c} v_0 & v_1 & \cdots & v_m \end{array} \right)$.
- This matrix is $m \times (m+1)$ and hence there exists a nontrivial x_n such that $Vx_n = 0$.

Theorem

Let S be a nontrivial subspace of $\mathbb{R}^m$. (In other words, $S \neq \{0\}$.)

Then there exists a basis $\{v_0, v_1, \ldots, v_{n-1}\} \subset \mathbb{R}^m$ such that $\mathrm{Span}(v_0, v_1, \ldots, v_{n-1}) = S$.

Proof:

Notice that we have already established that $m < n$. We will construct the vectors.

- Let S be a nontrivial subspace.
- Then S contains at least one nonzero vector.
- Let v_0 equal such a vector.
- Either $\mathrm{Span}(v_0) = S$ or $S \backslash \mathrm{Span}(v_0)$ is not empty. Pick $v_1 0$ in $S \backslash \mathrm{Span}(v_0)$.
- Either $\mathrm{Span}(v_0, v_1) = S$ or $S \backslash \mathrm{Span}(v_0, v_1)$ is not empty. Pick v_2 in $S \backslash \mathrm{Span}(v_0, v_1)$.

(2-2) dimension of a subspace

- 차원의 정의 : number of the basis ← dimension of a subspace S equals the number of vectors in a basis for that subspace

- dimension : The dimension of the column space is called the rank of the matrix. The rank is equal to the number of pivots in the reduced row echelon form, and is the maximum number of linearly independent columns that can be chosen from the matrix. <위키피디아>

- the number of vectors in a basis for a given subspace is always the same

Theorem

Let S be a subspace of $\mathbb{R}^m$ and let $\{v_0, v_1, \cdots, v_{n-1}\} \subset \mathbb{R}^m$ and $\{w_0, w_1, \cdots, w_{k-1}\} \subset \mathbb{R}^m$ both be bases for S.

Then $k = n$.

→ contradiction으로 증명해보자. k > n이라고 가정하자. wj=Vxj (즉, w vector를 v로 표현할 수 있다. basis이기 때문에)라고 하자. W=VX (X=(x0|x1|....|xk-1). X는 R^{nxk}에 속한다. 그런데 n보다 k가 크므로 X는 linearly dependent하고 Null space에 0이 아닌 vector를 해로 가진다. (그 해를 y라고 하자)

Wy = VXy = V(Xy) = V(0) = 0, 즉 Wy=0에서 y가 0이 아니므로 w는 linearly dependent하며 이는 basis라는 초기 가정에 모순이다.

Proof:

- W.l.o.g let us assume that $k > n$.
- Let $V = \left(v_0 \,\middle|\, \cdots \,\middle|\, v_{n-1} \right)$ and $W = \left(w_0 \,\middle|\, \cdots \,\middle|\, w_{k-1} \right)$.
- Let x_j have the property that $w_j = Vx_j$.
- Then $W = VX$, where $X = \left(x_0 \,\middle|\, \cdots \,\middle|\, x_{k-1} \right)$.
- $X \in \mathbb{R}^{n \times k}$ and recall that $k > n$.
- Let $y \neq 0 \in \mathcal{N}(X)$.
- $Wy = VXy = V(Xy) = V(0) = 0$.

(2-3) rank(A) : number of basis for the column space of A (dimension)

The following statements are equivalent statements about $A \in \mathbb{R}^{n \times n}$:

- A is nonsingular.
- A is invertible.
- A^{-1} exists.
- $AA^{-1} = A^{-1}A = I$.
- A represents a linear transformation that is a bijection.
- $Ax = b$ has a unique solution for all $b \in \mathbb{R}^n$.
- $Ax = 0$ implies that $x = 0$.
- $Ax = e_j$ has a solution for all $j \in \{0, \ldots, n-1\}$.
- The determinant of A is nonzero: $\det(A) \neq 0$.
- LU with partial pivoting does not break down.
- $\mathcal{C}(A) = \mathbb{R}^n$.
- A has linearly independent columns.
- $\mathcal{N}(A) = \{0\}$
- rank(A) = n.

(3) linear independence (linearly independent)

- 직관적 의미 : it means that we don't want to include vectors that we don't have to.
  - {x1, x2, x3, ..., xn} : 서로가 서로를 조합해서 x1 ~ xn을 만들어내지 못한다.

The set of vectors

$$\left\{\begin{pmatrix}1\\0\end{pmatrix},\begin{pmatrix}0\\1\end{pmatrix},\begin{pmatrix}1\\1\end{pmatrix}\right\}$$

is linearly dependent:

$$\begin{pmatrix}1\\0\end{pmatrix}+\begin{pmatrix}0\\1\end{pmatrix}-\begin{pmatrix}1\\1\end{pmatrix}=\begin{pmatrix}0\\0\end{pmatrix}.$$

- 수학적 정의 :

Definition

Let $\{v_0,\ldots,v_{n-1}\}\subset\mathbb{R}^m$.

This set of vectors is said to be *linearly independent* if

$$\chi_0 v_0+\chi_1 v_1+\cdots+\chi_{n-1}v_{n-1}=0$$

implies that

$$\chi_0=\cdots=\chi_{n-1}=0.$$

A set of vectors that is not linearly independent is said to be *linearly dependent*.

  - dependent :

Homework

Let the set of vectors $\{a_0, a_1, \ldots, a_{n-1}\} \subset \mathbb{R}^m$ be linearly dependent. Then at least one of these vectors can be written as a linear combination of the others.

True/False

True

There must exist $\chi_0, \chi_1, \ldots, \chi_{n-1} \in \mathbb{R}$ such that

$$\chi_0 a_0 + \chi_1 a_1 + \cdots + \chi_{n-1} a_{n-1} = 0$$

and for at least one j, $0 \leq j < n$, $\chi_j \neq 0$.

But then

$$\chi_j a_j = -\chi_0 a_0 - \chi_1 a_1 - \cdots - \chi_{j-1} a_{j-1} - \chi_{j+1} a_{j+1} - \cdots - \chi_{n-1} a_{n-1}$$

and therefore

$$a_j = -\frac{\chi_0}{\chi_j} a_0 - \frac{\chi_1}{\chi_j} a_1 - \cdots - \frac{\chi_{j-1}}{\chi_j} a_{j-1} - \frac{\chi_{j+1}}{\chi_j} a_{j+1} - \cdots - \frac{\chi_{n-1}}{\chi_j} a_{n-1}.$$

- Linearly independent 하다는 것의 판정 : Null space = 0 (Ax=0이 되게 하는 x의 집합)

Theorem

Let $\{a_0, \ldots, a_{n-1}\} \subset \mathbb{R}^m$ and let $A = \left(a_0 \,\middle|\, \cdots \,\middle|\, a_{n-1} \right)$. Then the vectors $\{a_0, \ldots, a_{n-1}\}$ are linearly independent if and only if $\mathcal{N}(A) = \{0\}$.

Proof:

[($\Rightarrow$)] Assume $\{a_0, \ldots, a_{n-1}\}$ are linearly independent. Need to show: $\mathcal{N}(A) = \{0\}$. Assume $x \in \mathcal{N}(A)$. Then $Ax = 0$ implies that

$$\begin{aligned} 0 &= Ax = \left(a_0 \,\middle|\, \cdots \,\middle|\, a_{n-1} \right) \begin{pmatrix} \chi_0 \\ \vdots \\ \chi_{n-1} \end{pmatrix} \\ &= \chi_0 a_0 + \chi_1 a_1 + \cdots + \chi_{n-1} a_{n-1}. \end{aligned}$$

Hence $x = 0$.

[($\Leftarrow$)] Want to prove $P \Leftarrow Q$, where P represents "the vectors $\{a_0, \ldots, a_{n-1}\}$ are linearly independent" and Q represents "$\mathcal{N}(A) = \{0\}$". Suffices to prove the **contrapositive**: $\neg P \Rightarrow \neg Q$. Assume that $\{a_0, \ldots, a_{n-1}\}$ are *not* linearly independent. Then there exist $\{\chi_0, \cdots, \chi_{n-1}\}$ with at least one $\chi_j \neq 0$ such that $\chi_0 a_0 + \chi_1 a_1 + \cdots + \chi_{n-1} a_{n-1} = 0$. Let $x = (\chi_0, \ldots, \chi_{n-1})^T$. Then $Ax = 0$ which means $x \in \mathcal{N}(A)$, and hence $\mathcal{N}(A) \neq \{0\}$.

- 예시1 :

The columns of an identity matrix $I \in \mathbb{R}^{n \times n}$ form a linearly independent set of vectors.

Proof: $\mathcal{N}(I) = \{0\}$. Thus, the columns of I are linearly independent.

- 예시2 : lower triangular matrix with nonzero on its diagonal도 마찬가지. column 벡타들이 linearly independent하다.

The columns of $L = \begin{pmatrix} 1 & 0 & 0 \\ 2 & -1 & 0 \\ 1 & 2 & 3 \end{pmatrix}$ are linearly independent.

$Lx = 0$

- 예시 3 : lower triangular matrix가 포함된 것.

The columns of $L = \begin{pmatrix} 1 & 0 & 0 \\ 2 & -1 & 0 \\ 1 & 2 & 3 \\ \hline -1 & 0 & -2 \end{pmatrix}$ are linearly independent.

l_1^T

$Lx = 0$ $\begin{pmatrix} L_0 \\ l_1^T \end{pmatrix} x = 0$ $L_0 x = 0$

- 예시 4 : free variable이 남는 경우에, Ax=0이 되도록 하더라도, 자유롭게 0이 아닌 xn을 집어넣을 수 있게 되기 때문에, linearly dependent가 된다.

Theorem

m n

Let $\{a_0, a_1, \ldots, a_{n-1}\} \in \mathbb{R}^m$ and $n > m$. Then these vectors are linearly dependent.

Proof:
Consider the matrix $A = \left(a_0 \,\middle|\, \cdots \,\middle|\, a_{n-1} \right)$. If one applies the Gauss-Jordan method to this matrix in order to get it to upper triangular form, at most m columns with pivots will be encountered. The other $n - m$ columns correspond to free variables, which allow us to construct nonzero vectors x so that $Ax = 0$.

- 역행렬과의 연관성

The following statements are equivalent statements about $A \in \mathbb{R}^{n \times n}$:

- A is nonsingular.
- A is invertible.
- A^{-1} exists.
- $AA^{-1} = A^{-1}A = I$.
- A represents a linear transformation that is a bijection.
- $Ax = b$ has a unique solution for all $b \in \mathbb{R}^n$.
- $Ax = 0$ implies that $x = 0$.
- $Ax = e_j$ has a solution for all $j \in \{0, \ldots, n-1\}$.
- The determinant of A is nonzero: $\det(A) \neq 0$.
- LU with partial pivoting does not break down.
- $\mathcal{C}(A) = \mathbb{R}^n$.
- A has linearly independent columns.
- $\mathcal{N}(A) = \{0\}$.

9.4.2. The column space

- Column space of A의 정의: C(A)라고도 한다.

Let $A \in \mathbb{R}^{m \times n}$. The **column space** of A equals the set

$$\{Ax \mid x \in \mathbb{R}^n\}.$$

- column space라고 부르는 이유 :

Why "Column Space"?

$$
\begin{aligned}
Ax &= \left(a_0 \,\middle|\, a_1 \,\middle|\, \cdots \,\middle|\, a_{n-1} \right) \begin{pmatrix} \chi_0 \\ \chi_1 \\ \vdots \\ \chi_{n-1} \end{pmatrix} \\
&= \chi_0 a_0 + \chi_1 a_1 + \cdots + \chi_{n-1} a_{n-1}.
\end{aligned}
$$

$\mathcal{C}(A)$ equals the set of all linear combinations of the columns of matrix A.

→ A의 column vector들의 linear combination의 집합이다.

- column space는 span과 개념이 같다. : 벡타의 linear combination으로 만드는 모든 조합

Three observations

- Given a set of vectors $\{v_0, v_1, \ldots, v_{n-1}\} \subset \mathbb{R}^n$, we can create

$$V = \left(v_0 \,\middle|\, v_1 \,\middle|\, \cdots \,\middle|\, v_{n-1} \right).$$

- Given a matrix $V \in \mathbb{R}^{m \times n}$ and vector $x \in \mathbb{R}^n$,

$$Vx = \chi_0 v_0 + \chi_1 v_1 + \cdots + \chi_{n-1} v_{n-1}.$$

- The column space of V, $\mathcal{C}(V)$:

$$
\begin{aligned}
\mathcal{C}(V) &= \{Vx \,|\, x \in \mathbb{R}^n\} \\
&= \{\chi_0 v_0 + \chi_1 v_1 + \cdots + \chi_{n-1} v_{n-1} \,|\, \chi_0, \chi_1, \ldots, \chi_{n-1} \in \mathbb{R}\}
\end{aligned}
$$

$$\text{Span}(v_0, v_1, \ldots, v_{n-1}) = \mathcal{C}(V)$$

- Column space의 특징 : subspace인가?

The set $S \subset \mathbb{R}^m$ described by $\{Ax \mid x \in \mathbb{R}^n\}$, where $A \in \mathbb{R}^{m \times n}$, is a subspace.

pf) 증명

(1) 영벡터 존재

(2) 덧셈

(3) 스칼라 곱

- $0 \in S$: (pick $x = 0$).
- If $v, w \in S$ then $(v + w) \in S$:
  Pick $v, w \in S$. Then for some $x, y \in \mathbb{R}^n$, $v = Ax$ and $w = Ay$. But then $v + w = Ax + Ay = A(x + y)$, which is also in S.
- If $\alpha \in \mathbb{R}$ and $v \in S$ then $\alpha v \in S$:
  Pick $\alpha \in \mathbb{R}$ and $v \in S$. Then for some $x \in \mathbb{R}^n$, $v = Ax$. But then $\alpha v = \alpha(Ax) = A(\alpha x)$, which is also in S since $\alpha x \in \mathbb{R}^n$.

- 앞의 내용과의 연결 : Ax=b는 언제 해를 갖는가?

$Ax = b$, has a solution if and only if b is in this space.

pf) 증명

Theorem

Let $A \in \mathbb{R}^{m \times n}$, $x \in \mathbb{R}^n$, and $b \in \mathbb{R}^m$. Then $Ax = b$ has a solution if and only if $b \in \mathcal{C}(A)$.

Proof

($\Rightarrow$) Assume that $Ax = b$. Then $b \in \{Ax | x \in \mathbb{R}^n\}$. Hence b is in the column space of A.

($\Leftarrow$) Assume that b is in the column space of A. Then $b \in \{Ax | x \in \mathbb{R}^n\}$. But this means there exists a vector x such that $Ax = b$.

- 결론 : column space는 무한개의 원소(벡터)를 가진 집합이다. 그런데 이는 A의 column 벡터의 합으로 모두 표현될 수 있다.

- While the set $\mathcal{C}(A)$ (typically) has an infinite number of elements, it can be completely described by the columns of A.

9.4.3. The null space

- 정의 :

Let $A \in \mathbb{R}^{m \times n}$. The set of all vectors $x \in \mathbb{R}^n$ that have the property that $Ax = 0$ is called the *null space* of A.

Notation:

$$\mathcal{N}(A) = \{x | Ax = 0\}.$$

Recall

- We are interested in the solutions of $Ax = b$.
- We have already seen that if $Ax_s = b$ and $Ax_n = 0$ then $x_s + x_n$ is also a solution:

$$A(x_s + x_n) = b.$$

- Hence, identifying the set of all vectors such that $Ax = 0$ is important to us.

- N(A)도 subspace인가?

Let $A \in \mathbb{R}^{m \times n}$.

The null space of A, $\mathcal{N}(A)$, is a subspace

True/False

- $0 \in \mathcal{N}(A)$: $A0 = 0$.
- If $x, y \in \mathcal{N}(A)$ then $x + y \in \mathcal{N}(A)$:
  Let $x, y \in \mathcal{N}(A)$ so that $Ax = 0$ and $Ay = 0$. Then $A(x + y) = Ax + Ay = 0 + 0 = 0$ which means that $x + y \in \mathcal{N}(A)$.
- If $\alpha \in \mathbb{R}$ and $x \in \mathcal{N}(A)$ then $\alpha x \in \mathcal{N}(A)$:
  Let $\alpha \in \mathbb{R}$ and $x \in \mathcal{N}(A)$ so that $Ax = 0$. Then $A(\alpha x) = A\alpha x = \alpha Ax = \alpha 0 = 0$ which means that $\alpha x \in \mathcal{N}(A)$.

10장.

10.1. Visualizing planes, lines, and solutions.

From the opener for Week 9:

$$\begin{array}{rcrcrcr} \chi_0 & - & 2\chi_1 & + & 4\chi_2 & = & -1 \\ \chi_0 & & & & & = & 2 \\ \chi_0 & + & 2\chi_1 & + & 4\chi_2 & = & 3 \end{array}$$

Solution:

$$\begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix} = \begin{pmatrix} 2 \\ 1 \\ -0.25 \end{pmatrix}$$

Important:

$$\begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \end{pmatrix} = \begin{pmatrix} 2 \\ 1 \\ -0.25 \end{pmatrix}$$

solves each of the individual equations.

- dependent variable, free variable 개념

10.1.1. 예시 1 : 식이 1개 일 때의 visualization

Visualize a plane of solutions corresponding to a single linear equation

$$\chi_0 - 2\chi_1 + 4\chi_2 = -1$$

As an appended system:

$$\left(\begin{array}{ccc|c} \boxed{1} & -2 & 4 & -1 \\ \uparrow & \uparrow & \uparrow & \end{array} \right).$$

dependent variable / free variable / free variable

→ free variable은 아무 value나 다 가질 수 있다. x0, x1, x2 중에서 아무거나 1개를 free variable로 정하면 다른 x1, x2는 그에 따라서 정해진다.

(1) 1단계 : Ax=b를 찾자.

Identify a specific solution

$$\chi_0 - 2\chi_1 + 4\chi_2 = -1$$

$$x_s = \begin{pmatrix} \chi_0 \\ 0 \\ 0 \end{pmatrix} \begin{matrix} \longleftarrow \text{Dependent variable to be computed} \\ \longleftarrow \text{Free variable} \\ \longleftarrow \text{Free variable} \end{matrix}$$

Substituting $\chi_1 = \chi_2 = 0$ into the equation gives us

$$\chi_0 - 2(0) + 4(0) = -1.$$

or $\chi_0 = -1$ so that

$$x_s = \begin{pmatrix} -1 \\ 0 \\ 0 \end{pmatrix}.$$

(2) 2단계 : Ax=0인 x를 찾자.

Identify linearly independent vectors in the null space

$$\chi_0 - 2\chi_1 + 4\chi_2 = 0.$$

We look for solutions in the form

$$x_{n_0} = \begin{pmatrix} \chi_0 \\ 1 \\ 0 \end{pmatrix} \begin{matrix} \longleftarrow \text{Dependent variable to be computed} \\ \longleftarrow \text{Free variable} \\ \longleftarrow \text{Free variable} \end{matrix}$$

and

$$x_{n_1} = \begin{pmatrix} \chi_0 \\ 0 \\ 1 \end{pmatrix} \begin{matrix} \longleftarrow \text{Dependent variable to be computed} \\ \longleftarrow \text{Free variable} \\ \longleftarrow \text{Free variable} \end{matrix}$$

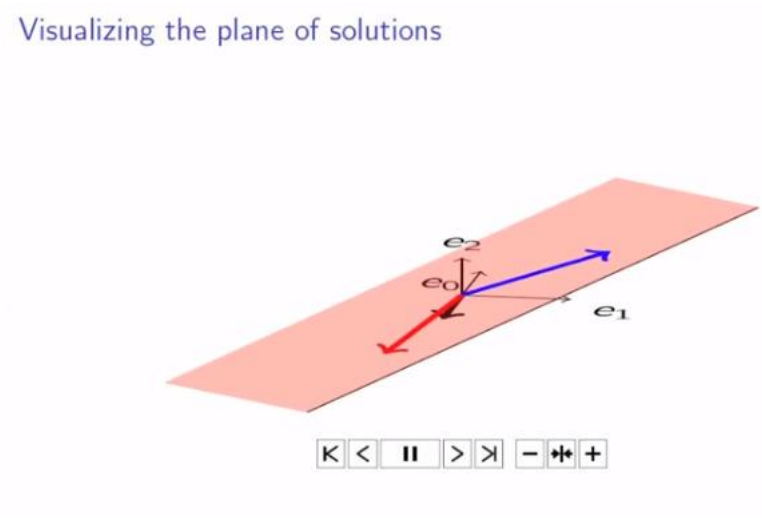
(3) 3단계 : general solution 도출 = specific solution + zero형

Identify linearly independent vectors in the null space

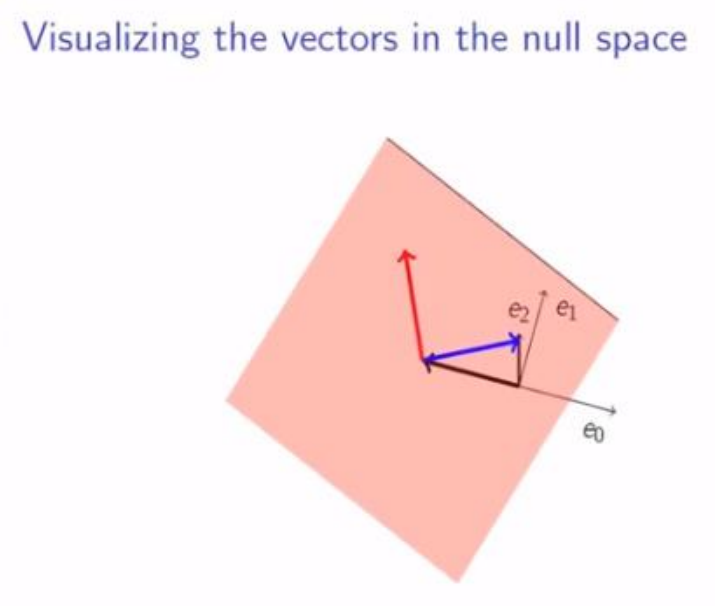
Thus two (nonunique) linearly independent vectors in the null space are given by

$$x_{n_0} = \begin{pmatrix} 2 \\ 1 \\ 0 \end{pmatrix} \quad \text{and} \quad x_{n_1} = \begin{pmatrix} -4 \\ 0 \\ 1 \end{pmatrix}.$$

$$x_{\text{general}} = x_s + \alpha x_{n_0} + \beta x_{n_1}$$



→ Null solution is shifted by specific solution. zero로 mapping되는 vector가 수직으로 설정하면, 더 편할 수 있다.



→ 11장에서 수직으로 나오는 방법을 배운다.

다른 예 :

Identify linearly independent vectors in the null space

$$\chi_0 + (0)\chi_1 + (0)\chi_2 = 0.$$

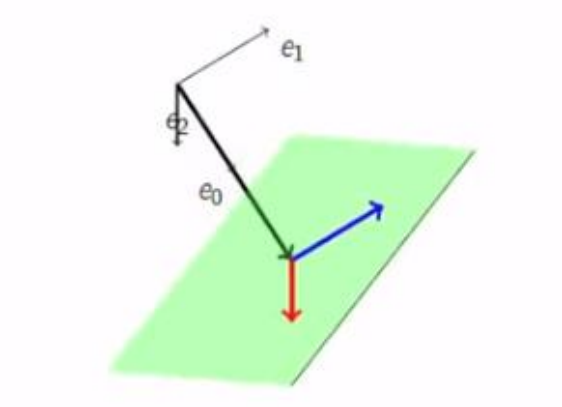
We look for solutions in the form

$$x_{n_0} = \begin{pmatrix} \chi_0 \\ 1 \\ 0 \end{pmatrix} \begin{array}{l} \longleftarrow \text{Dependent variable to be computed} \\ \longleftarrow \text{Free variable} \\ \longleftarrow \text{Free variable} \end{array}$$

and

$$x_{n_1} = \begin{pmatrix} \chi_0 \\ 0 \\ 1 \end{pmatrix} \begin{array}{l} \longleftarrow \text{Dependent variable to be computed} \\ \longleftarrow \text{Free variable} \\ \longleftarrow \text{Free variable} \end{array}$$

→ perpendicular하게!!



10.1.2. 예시2: 식이 2개일 때 visualization

$$\begin{array}{rcrcrcr} \chi_0 & - & 2\chi_1 & + & 4\chi_2 & = & -1 \\ \chi_0 & + & (0)\chi_1 & + & (0)\chi_2 & = & 2 \end{array}$$

Appended system:

$$\left(\begin{array}{ccc|c} 1 & -2 & 4 & -1 \\ 1 & 0 & 0 & 2 \end{array}\right)$$

$$\left(\begin{array}{ccc|c} 1 & -2 & 4 & -1 \\ 1 & 0 & 0 & 2 \end{array}\right)$$

$$\downarrow$$

$$\left(\begin{array}{ccc|c} 1 & -2 & 4 & -1 \\ & 2 & -4 & 3 \end{array}\right)$$

dependent variable ↑ dependent variable ↑ ↑ free variable

Adding lines where the planes meet

We know a specific solution since we know the solution to the three equations in three unknowns.

Find vector in null space:

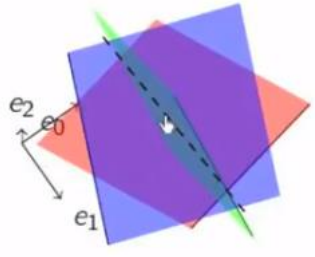
$$\left(\begin{array}{ccc|c} 1 & -2 & 4 & 0 \\ & 2 & -4 & 0 \end{array}\right)$$

$$x_n = \begin{pmatrix} \chi_0 \\ \chi_1 \\ 1 \end{pmatrix}$$

$$\begin{array}{rrcr} \chi_0 & -2\chi_1 & = & -4 \\ & 2\chi_1 & = & 4 \end{array}$$

$$x_n = \begin{pmatrix} 0 \\ 2 \\ 1 \end{pmatrix}$$

→ 두 평면의 교차되는 직선을 찾는 과정이다.



Adding lines where the planes meet

We know a specific solution since we know the solution to the three equations in three unknowns.

Find vector in null space:

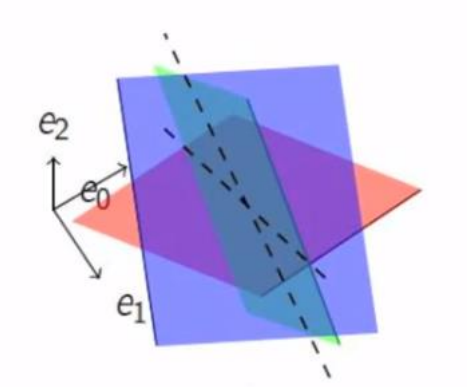
$$\left(\begin{array}{ccc|c} 1 & 0 & 0 & 0 \\ 0 & 2 & 4 & 0 \end{array}\right)$$

$$x_n = \begin{pmatrix} \chi_0 \\ \chi_1 \\ 1 \end{pmatrix}$$

$$\begin{array}{rcl} \chi_0 & = & 0 \\ 2\chi_2 & = & -4 \end{array}$$

$$x_n = \begin{pmatrix} 0 \\ -2 \\ 1 \end{pmatrix}$$

→ 다른 두 식도 똑같이 하면 선 하나를 더 그릴 수 있다.



한번 더!

Adding lines where the planes meet

We know a specific solution since we know the solution to the three equations in three unknowns.

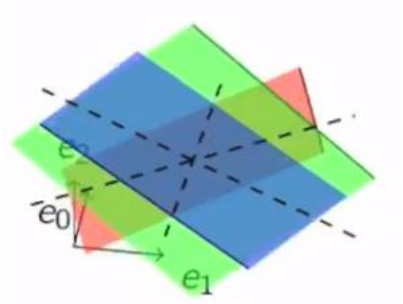
Find vector in null space:

$$\left(\begin{array}{ccc|c} 1 & -2 & 4 & 0 \\ 0 & 4 & 0 & 0 \end{array}\right)$$

$$x_n = \begin{pmatrix} \chi_0 \\ \chi_1 \\ 1 \end{pmatrix}$$

$$\begin{array}{rcl} \chi_0 - 2\chi_1 & = & -4 \\ 4\chi_1 & = & 0 \end{array}$$

$$x_n = \begin{pmatrix} -4 \\ 0 \\ 1 \end{pmatrix}$$



뭐 거의 이해를 못하겠는데??

10.2. 용어 정리

- Row-echelon form of the system

→ 행렬이 upper triangular matrix와 유사하게 되도록 정리한 형태

$$\underbrace{\begin{pmatrix} 1 & 3 & 1 & 2 \\ 2 & 6 & 4 & 8 \\ 0 & 0 & 2 & 4 \end{pmatrix}}_{A} \begin{pmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 1 \\ 3 \\ 1 \end{pmatrix} \longrightarrow \left(\begin{array}{cccc|c} 1 & 3 & 1 & 2 & 1 \\ 2 & 6 & 4 & 8 & 3 \\ 0 & 0 & 2 & 4 & 1 \end{array}\right)$$

$$\longrightarrow \left(\begin{array}{cccc|c} \boxed{1} & 3 & 1 & 2 & 1 \\ 0 & 0 & \boxed{2} & 4 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{array}\right)$$

- Pivot : first non-zero starting from the left.

$$\left(\begin{array}{cccc|c} \boxed{1} & 3 & 1 & 2 & 1 \\ 0 & 0 & \boxed{2} & 4 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{array}\right)$$

- Free variable : pivot항이 포함되지 않은 변수, 여기서는 X1, X3이다.

- Dependent variable : free variable이 아닌 변수

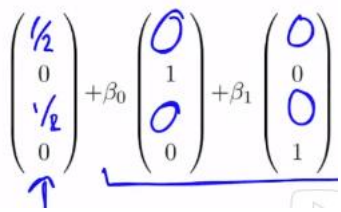
- Specific solution : free variable이 포함된 변수를 0이라고 하고 specific variable을 구해보자. 그러면 다른 2개는 정해지게 된다.

- Basis for null space : Ax=b에서 b를 0이라고 하고서, free variable이 (0,0)이 아닌 해 벡터를 구한다. 여기서는 1, 0 / 0, 1라고 하고 두 개의 벡터를 구했다. 두 개의 벡터를 통해서 모든 해 벡터를 표현할 수 있을 것이다.

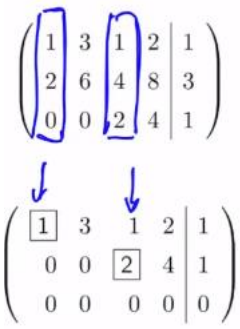
$$\left(\begin{array}{cccc|c} \boxed{1} & 3 & 1 & 2 & 0 \\ 0 & 0 & \boxed{2} & 4 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{array}\right)$$

$$\begin{pmatrix} \square \\ 1 \\ \square \\ 0 \end{pmatrix} \text{ and } \begin{pmatrix} \square \\ 0 \\ \square \\ 1 \end{pmatrix}$$

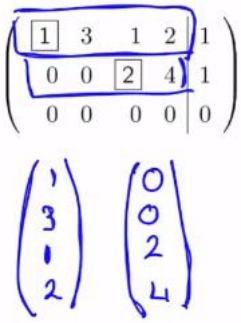
- general solution = specific solution + linear combination of the basis for null space



- a basis for column space (이것은 못알아듣겠다)
  - pivot이 생기는 열의 각 열이 바로 basis of column space이다. (왜???)



- a basis for row space (이것은 못알아듣겠다)
  - pivot이 있는 행의 각 행을 열로 linear combination을 만든 것이 바로 basis of row space이다. (왜???)



The row space is the subspace of all vectors that can be created by taking linear combinations of the rows of a matrix. In other words, the row space of equals (the column space of).

- Dimension of row space, dimension of column space, rank of matrix
  ➔ 모두 다 pivot의 개수(k)의 값이다. 이것이 바로 basis vector의 개수이다.
- dimension of the null space = n-k

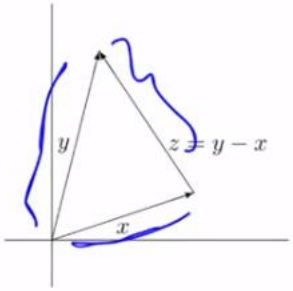
10.3. Orthogonal vector

- 조건 : $x^Ty = 0$ 이면 x, y 벡터는 직교한다.

- 증명 :

If $x, y \in \mathbb{R}^n$ are linearly independent.

- $\text{Span}(\{x, y\}$ forms a 2D subspace.
- x, y, and $(x - y)$ lie in this 2D subspace.
- They form a triangle:



- x and y are orthogonal if they meet at a right angle.
- Pythagorean Theorem *if* the angle in the angle where x and y meet is a right angle, then $\|z\|_2^2 = \|x\|_2^2 + \|y\|_2^2$.

→ x-y로 linear combination이므로 같은 2D 평면 안에 있게 된다. 그리고 삼각형을 이룬다. 피타고라스 법칙을 적용하면 증명을 할 수 있다.

- $$\begin{aligned}
\|z\|_2^2 = \|x\|_2^2 + \|y\|_2^2 &= \|y - x\|_2^2 \\
&= (y - x)^T (y - x) \\
&= (y^T - x^T)(y - x) \\
&= (y^T - x^T)y - (y^T - x^T)x \\
&= \underbrace{y^T y}_{\|y\|_2^2} - \underbrace{(x^T y + y^T x)}_{2x^T y} + \underbrace{x^T x}_{\|x\|_2^2} \\
&= \|x\|_2^2 - 2x^T y + \|y\|_2^2.
\end{aligned}$$

- When x and y are perpendicular (orthogonal)
$$\|x\|_2^2 + \|y\|_2^2 = \|x\|_2^2 - 2x^T y + \|y\|_2^2.$$
- Cancelling terms on the left and right of the equality, this implies that $x^T y = 0$.

10.3.1. Orthogonal space

- 정의 : V 내의 모든 vector v는 W내의 모든 벡터 w와 직교해야 한다. 그러면 V와 W를 orthogonal한 subspace라고 할 수 있다.

Let $\mathbf{V}, \mathbf{W} \subset \mathbb{R}^n$ be subspaces. $\mathbf{V}$ and $\mathbf{W}$ are said to be orthogonal if

$$v \in \mathbf{V} \text{ and } w \in \mathbf{W} \text{ implies that } v^T w = 0.$$

Notation: $\mathbf{V} \perp \mathbf{W}$

- 표기 : V⊥ ㅋㅋㅋㅋ

Given subspace $\mathbf{V} \subset \mathbb{R}^n$, the set of all vectors in $\mathbb{R}^n$ that are orthogonal to $\mathbf{V}$ is denoted by $\mathbf{V}^\perp$ (pronounced as "V-perp").

- V가 subspace이면, V⊥도 subspace일까? 맞다

  - 증명 : cf) subspace란 (1) 0 벡터를 포함하고 (2) 덧셈 (3) 스칼라 곱에 대해서 닫혀 있어야 한다.

Homework

If $\mathbf{V} \in \mathbb{R}^n$ is a subspace, then $\mathbf{V}^\perp$ is a subspace.

True/False

(1) 0벡터를 포함하는가.

- $0 \in \mathbf{V}^\perp$: Let $x \in \mathbf{V}$. Then $0^T x = 0$ and hence $0 \in \mathbf{V}^\perp$.

(2) 덧셈에 대해서 닫혀 있는가

x, y ∈ V⊥이면, x+y ∈ V⊥ 라는 것을 증명해야 한다.

- If $x, y \in \mathbf{V}^\perp$ then $x + y \in \mathbf{V}^\perp$: Let $x, y \in \mathbf{V}^\perp$ and let $z \in \mathbf{V}$. We need to show that $(x + y)^T z = 0$.

$$
\begin{aligned}
& (x+y)^T z \\
& \quad = \quad < \text{property of dot} > \\
& x^T z + y^T z \\
& \quad = \quad < x, y \in \mathbf{V}^\perp \text{ and } z \in \mathbf{V} > \\
& 0 + 0 \\
& \quad = \quad < algebra > \\
& 0
\end{aligned}
$$

Hence $x + y \in \mathbf{V}^\perp$.

(3) 스칼라곱 증명

x ∈ V⊥이면, αx ∈ V⊥라는 것을 증명해야 한다.

- If $\alpha \in \mathbb{R}$ and $x \in \mathbf{V}^{\perp}$ then $\alpha x \in \mathbf{V}^{\perp}$: Let $\alpha \in \mathbb{R}$, $x \in \mathbf{V}^{\perp}$ and let $z \in \mathbf{V}$. We need to show that $(\alpha x)^T z = 0$.

$$
\begin{aligned}
&(\alpha x)^T z \\
&\quad = \quad < \text{algebra} > \\
&\alpha x^T z \\
&\quad = \quad < x \in \mathbf{V}^{\perp} \text{ and } z \in \mathbf{V} > \\
&\alpha 0 \\
&\quad = \quad < \text{algebra} > \\
&0
\end{aligned}
$$

Hence $\alpha x \in \mathbf{V}^{\perp}$.

10.4. Four fundamental spaces

- 종류

Given matrix $A \in \mathbb{R}^{m \times n}$:

- Column space: $\mathcal{C}(A) \subset \mathbb{R}^m$.
- Null space: $\mathcal{N}(A) \subset \mathbb{R}^n$.

- Row space: $\mathcal{R}(A) \subset \mathbb{R}^n$ or $\mathcal{C}(A^T) \subset \mathbb{R}^n$.
- "Left null space": $\mathcal{N}(A^T) \subset \mathbb{R}^m$.

- 각 space의 항의 개수 : column space는 Ax이니까 (m x n) * (n x 1) → m차원이다.
null space의 경우에는 Ax=0의 해니까 (n x 1)이다.
left null space는

A^T*x를 하는 것이 A에다가 left에서 곱하는 거랑 같다고 보기 때문에 left라는 말을 쓴 듯 하다.

- 각 space의 차원(dimension)
  dim (C(A)) = k
  dim (N(A)) = n - k
  dim (R(A)) = dim (C(A^T)) = k
  dim (N(A^T)) = m - k

- Then there are $n-k$ free variables, corresponding to the columns in which no pivots reside. **This means that the null space dimension equals $n-k$**

????

- (증명) Row space와 Null space의 관계: orthogonal space

R(A) ⊥ N(A)

Theorem

Let $A \in \mathbb{R}^{m \times n}$. Then $\mathcal{R}(A) \perp \mathcal{N}(A)$.

Proof: Let $y \in \mathcal{R}(A)$ and $z \in \mathcal{N}(A)$. We need to prove that $y^T z = 0$.

$$
\begin{aligned}
& y^T z \\
&= \quad < y \in \mathcal{R}(A) \text{ implies that } y = A^T x \text{ for some } x \in \mathbb{R}^m >
\end{aligned}
$$

→ y = C(A^T)로 쓸 수 있으니까 어떤 x에 대해서 linear combination이 있다는 것을 알 수 있다.

$$
\begin{aligned}
& (A^T x)^T z \\
&= \quad < (AB)^T = B^T A^T \text{ and } (A^T)^T = A > \\
& x^T A z \\
&= \quad < z \in \mathcal{N}(A) \text{ implies that } Az = 0 > \\
& x^T 0 \\
&= \quad < \text{algebra} > \\
& 0
\end{aligned}
$$

→ 여기서 한발자국 더 나아가면, R(A) = C(A^T) ⊥ N(A)이다.

따라서 B = A^{T}를 대입하면 C(A) ⊥ N(A^T)이다. 따라서 column space와 left null space도 직교하는 것을 알 수 있다.

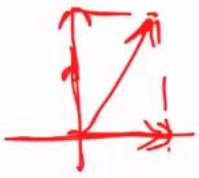
- (증명) 모든 vector x를 row space의 vector와 null space의 vector의 합으로 표현할 수 있다.

Theorem

Let $A \in \mathbb{R}^{m \times n}$. Every $x \in \mathbb{R}^n$ can be written as

$$x = x_r + x_n$$

where $x_r \in \mathcal{R}(A)$ and $x_n \in \mathcal{N}(A)$.

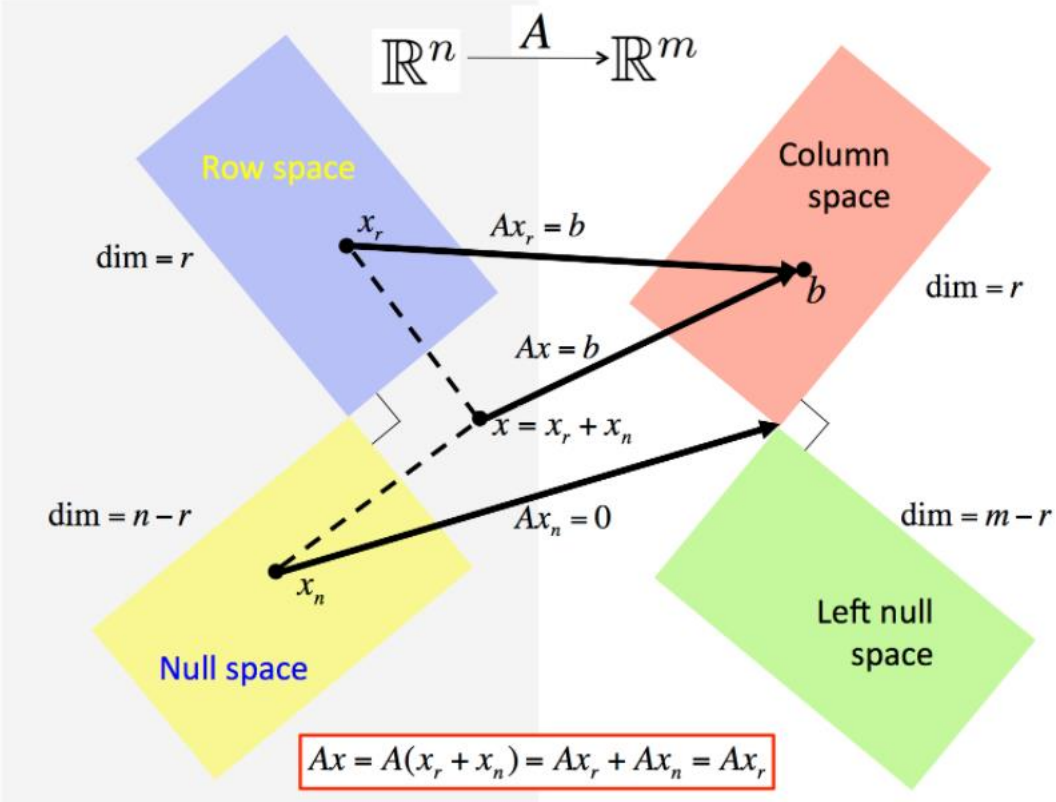


Proof:

- If $\dim(\mathcal{R}(A) = k$, then $\dim(\mathcal{N}(A)) = n - k$.
- $\{v_0, \ldots, v_{k-1}\}$ be a basis for $\mathcal{R}(A)$.
- $\{v_k, \ldots, v_{n-1}\}$ be a basis for $\mathcal{N}(A)$.
- Then $\{v_0, \ldots, v_{n-1}\}$ are linearly independent (and form a basis for $\mathbb{R}^n$). Details beyond this course.

$$x = \underbrace{\sum_{i=0}^{k-1} \beta_i v_i}_{x_r \in \mathcal{R}(A)} + \underbrace{\sum_{i=k}^{n-1} \beta_i v_i}_{x_n \in \mathcal{N}(A)}$$

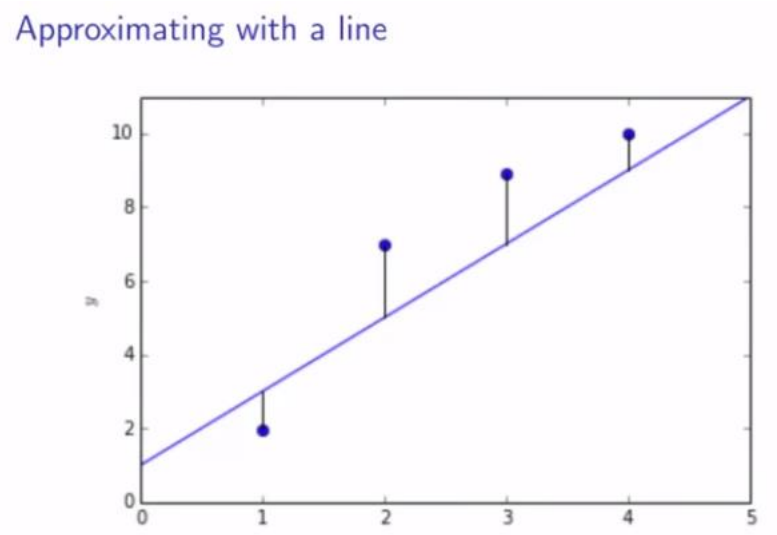
→ basis을 합하면 R^n을 모두 표현할 수 있게 되며 basis는 서로 linearly independent하다. 따라서 row space와 null space의 합으로 모든 벡타 x를 표현할 수 있다.



→ $x = x_r + x_n$은 위에서 증명했고, $Ax_n = 0$ (null space)이므로 $Ax = Ax_r = b$ 이다.

- 이 그림의 의의 : Ax=b가 해가 없을 때 어떻게 해야하는지를 알려준다.

10.5. linear least squares approximation = Normal Equation!!! 대박!!



- 행렬과 벡터로 이 회귀선을 추정하는 식을 표현해보자.

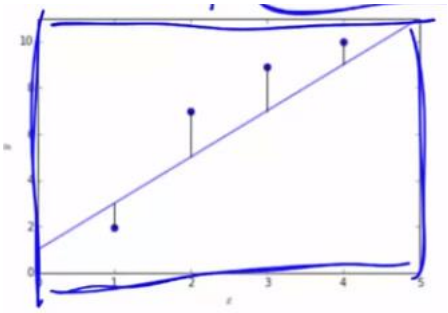
Using matrices and vectors

$$x = \begin{pmatrix} \chi_0 \\ \chi_1 \\ \chi_2 \\ \chi_3 \end{pmatrix} = \begin{pmatrix} 1 \\ 2 \\ 3 \\ 4 \end{pmatrix} \quad \text{and} \quad y = \begin{pmatrix} \psi_0 \\ \psi_1 \\ \psi_2 \\ \psi_3 \end{pmatrix} = \begin{pmatrix} 1.97 \\ 6.97 \\ 8.89 \\ 10.01 \end{pmatrix}.$$

$$y = \gamma_0 + \gamma_1 x$$

$$\begin{aligned} \psi_0 &= \gamma_0 + \gamma_1 \chi_0 \\ \psi_1 &= \gamma_0 + \gamma_1 \chi_1 \\ \psi_2 &= \gamma_0 + \gamma_1 \chi_2 \\ \psi_3 &= \gamma_0 + \gamma_1 \chi_3 \end{aligned} \quad \text{or, specifically,} \quad \begin{aligned} 1.97 &= \gamma_0 + \gamma_1 \\ 6.97 &= \gamma_0 + 2\gamma_1 \\ 8.89 &= \gamma_0 + 3\gamma_1 \\ 10.01 &= \gamma_0 + 4\gamma_1 \end{aligned}$$

$$\begin{pmatrix} \psi_0 \\ \psi_1 \\ \psi_2 \\ \psi_3 \end{pmatrix} = \begin{pmatrix} 1 & \chi_0 \\ 1 & \chi_1 \\ 1 & \chi_2 \\ 1 & \chi_3 \end{pmatrix} \begin{pmatrix} \gamma_0 \\ \gamma_1 \end{pmatrix} \quad \text{or,} \quad \begin{pmatrix} 1.97 \\ 6.97 \\ 8.89 \\ 10.01 \end{pmatrix} \approx \begin{pmatrix} 1 & 1 \\ 1 & 2 \\ 1 & 3 \\ 1 & 4 \end{pmatrix} \begin{pmatrix} \gamma_0 \\ \gamma_1 \end{pmatrix}$$



→ (내가 수업시간에 질문했던 것이 여기서 나오네!!!) 아 여기서 항등식에 의한 정확한 값을 구하기는 불가능하다. 이 네 개 점을 한꺼번에 지나가는 점을 찾을 수는 없기 때문이다. 따라서 이것은 항등식이 아니라, 근사식이 된다. Then how do we find the best coefficience??

→ 문제의 재정의 :

Find the best APPROXIMATE solution

$$Ax = b$$

$$b \notin \mathcal{C}(A)$$

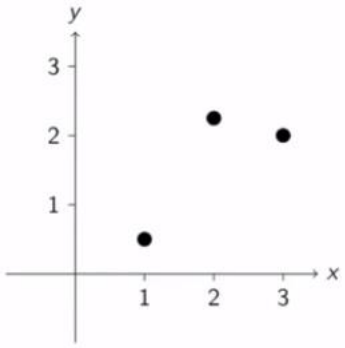
Find x so that Ax is as close to b as possible.

10.5.2. 간단한 예시와 visualization

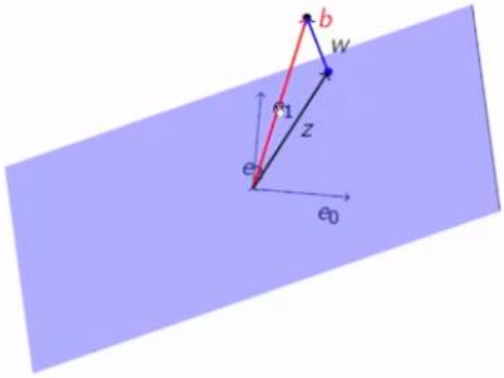
- Ax = b로 표현한 것. 주어진 점은 (1, 0.5), (2, 2.25), (3, 2)이다.

Simpler example yet

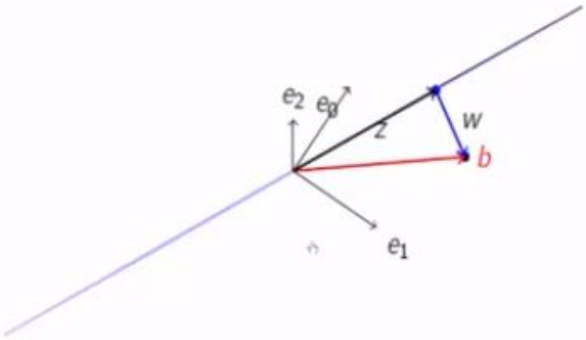
$$\begin{pmatrix} 1 & 1 \\ 1 & 2 \\ 1 & 3 \end{pmatrix} \begin{pmatrix} \gamma_0 \\ \gamma_1 \end{pmatrix} = \begin{pmatrix} 0.5 \\ 2.25 \\ 2 \end{pmatrix}$$



$$\text{Span}\left(\begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}, \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix}\right) \quad \text{and} \quad b = \begin{pmatrix} 0.5 \\ 2.25 \\ 2 \end{pmatrix}$$



→ 보라색은 행렬의 A의 column space이다. 즉 A의 열 들의 span(linear combination)이다. 빨간색은 b의 벡터이다. 그런데 b는 column space에 없다. column space에 있는 벡터랑 그것에 수직인 w를 합하면 b를 얻을 수 있다. (b = z + w)

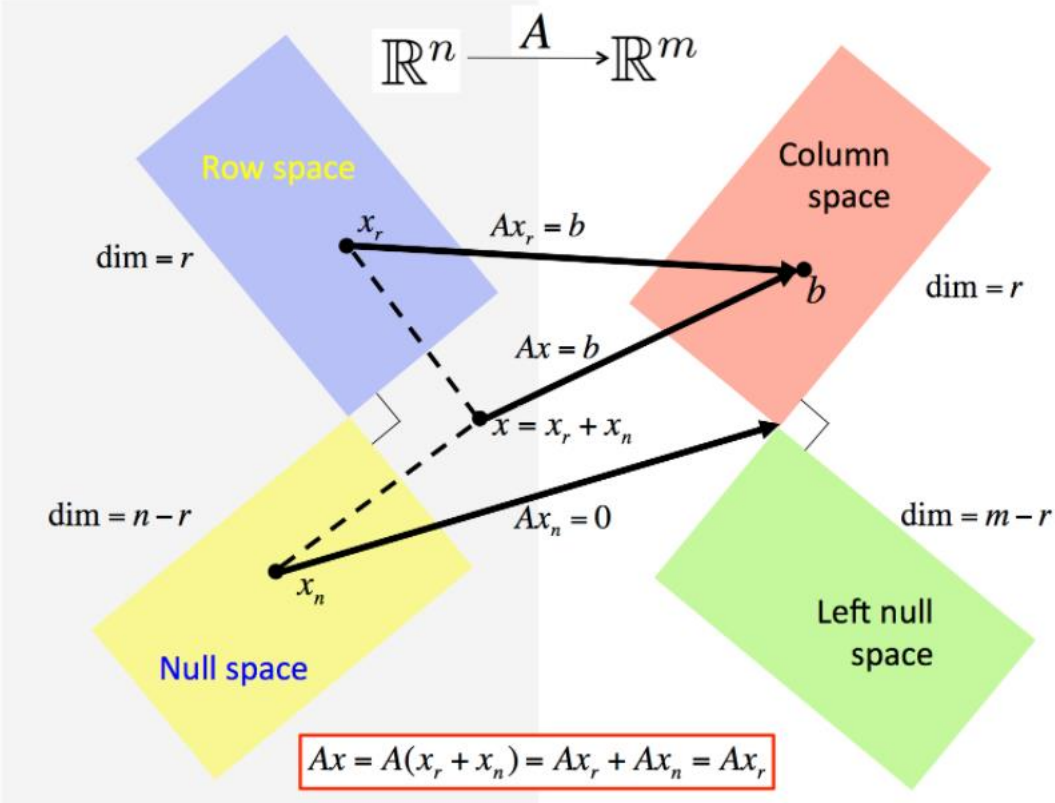


- 풀이 :

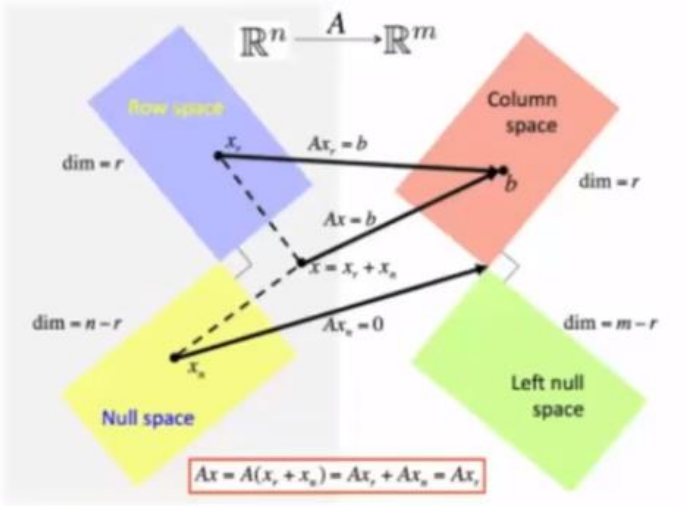
Best solution

- $Ax \approx b$
- Let $\hat{x}$ be best solution.
- $b = z + w$, where
  - $z \in \mathcal{C}(A)$; and
  - $w \in \mathcal{C}(A)^{\perp}$.
- $w \in \mathcal{C}(A)^{\perp} \Rightarrow$
  $w \in \mathcal{N}(A^T) \Rightarrow$
  $A^T w = 0.$

→ w는 A의 column space의 orthogonal space에 있다. C(A)ㅗ로 표현할 수 있다. 그런데 밑의 그림에 따르면 C(A)ㅗ는 바로 left null space이고 $N(A^T)$로 표현할 수 있다. 따라서, $A^T w=0$인 것이다.



- $b = z + w$, where
  - $z \in \mathcal{C}(A)$; and
  - $w \in \mathcal{C}(A)^{\perp}$.
- $w \in \mathcal{C}(A)^{\perp} \Rightarrow$ $w \in \mathcal{N}(A^T) \Rightarrow$ $A^T w = 0$.
- $0 = A^T \underbrace{(b - z)}_{w}$.
- $A^T(b - \underbrace{A\hat{x}}_{z}) = 0$.
- $A^T A\hat{x} = A^T b$.



→ $A^T w=0$을 다르게 표현하면 이와 같이 표현할 수 있다. $A\hat{x} = z$이다. 여기서 x hat는 best approximation이다.

이 식을 푸는 것이 바로 바로 **Normal equation**이다. 대박!!!

- $A^T A\hat{x} = A^T b$
- $A^T A$ is nonsingular if A has linearly independent columns.

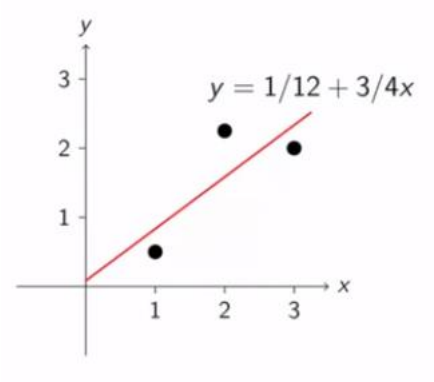
→ $A^T A$가 역행렬이 있으면 된다.

- 이것을 통해서 x hat을 구할 수 있다.

$$\underbrace{\begin{pmatrix} 1 & 1 \\ 1 & 2 \\ 1 & 3 \end{pmatrix}}_{A} \underbrace{\begin{pmatrix} \gamma_0 \\ \gamma_1 \end{pmatrix}}_{x} \approx \underbrace{\begin{pmatrix} 0.5 \\ 2.25 \\ 2 \end{pmatrix}}_{b}$$

Solve $A^T Ax = A^T b$:

- $A^T b = \begin{pmatrix} 1 & 1 \\ 1 & 2 \\ 1 & 3 \end{pmatrix}^T \begin{pmatrix} 0.5 \\ 2.25 \\ 2 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 2 & 3 \end{pmatrix} \begin{pmatrix} 0.5 \\ 2.25 \\ 2 \end{pmatrix} = \begin{pmatrix} 4.75 \\ 11 \end{pmatrix}$
- $A^T A = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 2 & 3 \end{pmatrix} \begin{pmatrix} 1 & 1 \\ 1 & 2 \\ 1 & 3 \end{pmatrix} = \begin{pmatrix} 3 & 6 \\ 6 & 14 \end{pmatrix}$.
- $(A^T A)^{-1} = \begin{pmatrix} 3 & 6 \\ 6 & 14 \end{pmatrix}^{-1} = \frac{1}{(3)(14)-(6)(6)} \begin{pmatrix} 14 & -6 \\ -6 & 3 \end{pmatrix} = \frac{1}{6} \begin{pmatrix} 14 & -6 \\ -6 & 3 \end{pmatrix}$.
- $x = (A^T A)^{-1} A^T b = \frac{1}{6} \begin{pmatrix} 14 & -6 \\ -6 & 3 \end{pmatrix} \begin{pmatrix} 4.75 \\ 11 \end{pmatrix} = \begin{pmatrix} 1/12 \\ 3/4 \end{pmatrix}$.



$x = (A^TA)^{-1}A^Tb$

- 정리 :

To solve $Ax \approx b$:

- $\widehat{b}$ is the vector in $C(A)$ closest to b.
- Instead solve $Ax = \widehat{b}$.
- x solves $A^TAx = A^Tb$.
- $x = (A^TA)^{-1}A^Tb$.

→ b는 column space에 없으니까 z = b hat을 구해야 한다.

→ 8장에 따르면 matrix를 invert하지 말자고 했다. $A^TAx = A^Tb$에서 (1) LU factorization을 하거나 (2) Cholesky factorization for symmetric positive definite matrices

- $\widehat{b} = Ax = A[(A^TA)^{-1}A^Tb]$: (orthogonal) projection of b onto $C(A)$.

→ b hat는 b의 column space에의 orthogonal projection이라고 볼 수 있다.

- 기억하는 방법 :

➔ A x = b 라고 써 놓고, 양변에 A^T를 곱한다고 생각해보자.

- 다음주와의 관련성 : orthogonal projection onto the column space of a matrix → low rank approximation을 할 것이다. → data compression을 할 수 있다.

- 위의 과정이 linear least square인 수학적 이유:

Equivalent statements:

- Find the "best" solution, $\widehat{x}$, to $Ax = b$.
- Find $\widehat{x}$ so that $A\widehat{x}$ is the (orthogonal) projection of b onto $\mathcal{C}(A)$.
- Find $\widehat{x}$ that minimizes the length of $b - Ax$:*

$$\|b - A\widehat{x}\|_2 = \min_x \|b - Ax\|_2.$$

→ 위에서의 w = b - z의 길이를 최소화하려는 것과 같다.

Find $\widehat{x}$ that minimizes the length of $b - Ax$:

$$\|b - A\widehat{x}\|_2 = \min_x \|b - Ax\|_2.$$

is equivalent to

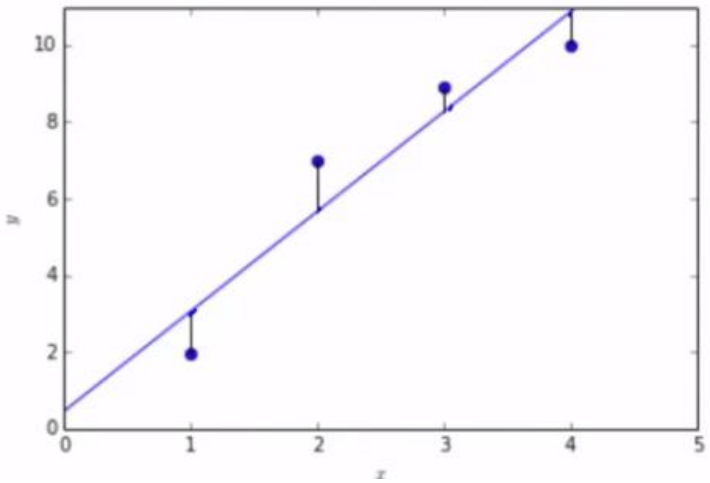
$$\|b - \underbrace{A\widehat{x}}_{y}\|_2^2 = \min_x \|b - Ax\|_2^2.$$

$$\|b - y\|_2^2 = \left\| \begin{pmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_{m-1} \end{pmatrix} - \begin{pmatrix} \psi_0 \\ \psi_1 \\ \vdots \\ \psi_{m-1} \end{pmatrix} \right\|_2^2 = \left\| \begin{pmatrix} \beta_0 - \psi_0 \\ \beta_1 - \psi_1 \\ \vdots \\ \beta_{m-1} - \psi_{m-1} \end{pmatrix} \right\|_2^2$$

$$= (\beta_0 - \psi_0)^2 + (\beta_1 - \psi_1)^2 + \cdots + (\beta_{m-1} - \psi_{m-1})^2$$

→ 양변에 제곱을 하면 위와 같다.

$$\|b - \widehat{b}\|_2^2 = (\beta_0 - \psi_0)^2 + (\beta_1 - \psi_1)^2 + \cdots + (\beta_{m-1} - \psi_{m-1})^2$$



- normal equation을 푸는 방법

10.5.1 Solving the Normal Equations

In our examples and exercises, we solved the normal equations

$$A^T Ax = A^T b,$$

where $A \in \mathbb{R}^{m \times n}$ has linear independent columns, via the following steps:

- Form $y = A^T b$
- Form $A^T A$.
- Invert $A^T A$ to compute $B = (A^T A)^{-1}$.
- Compute $\hat{x} = By = (A^T A)^{-1} A^T b$.

This involves the inversion of a matrix, and we claimed in Week 8 that one should (almost) never, ever invert a matrix.

In practice, this is not how it is done for larger systems of equations. Instead, one uses either the Cholesky factorization (which was discussed in the enrichment for Week 8), the QR factorization (to be discussed in Week 11), or the Singular Value Decomposition (SVD, which is briefly mentioned in Week 11).

Let us focus on how to use the Cholesky factorization. Here are the steps:

- Compute $C = A^T A$.
- Compute the Cholesky factorization $C = LL^T$, where L is lower triangular.
  This allows us to take advantage of symmetry in C.
- Compute $y = A^T b$.
- Solve $Lz = y$.
- Solve $L^T \hat{x} = z$.

The vector $\hat{x}$ is then the best solution (in the linear least-squares sense) to $Ax \approx b$.

The Cholesky factorization of a matrix, C, exists if and only if C has a special property. Namely, it must be symmetric positive definite (SPD).

→ 역행렬로 안하고 $A^TA\ x = A^Tb$ 푸는 방법 :

(1) Cholesky factorization ← LU factorization이랑 비슷하다.

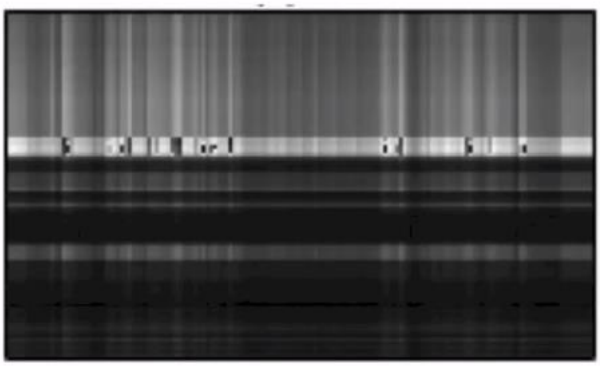
(2) QR factorization

(3) Singular Value Decomposition (SVD)

11장. Low rank approximation

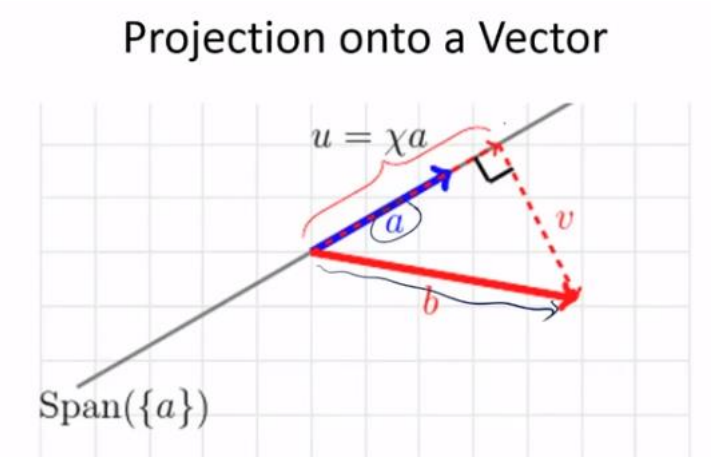
11.1. Low rank approximation

projection onto the column space of a matrix can be used to do data compression

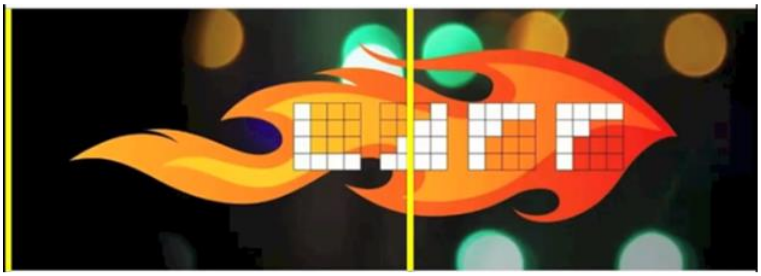


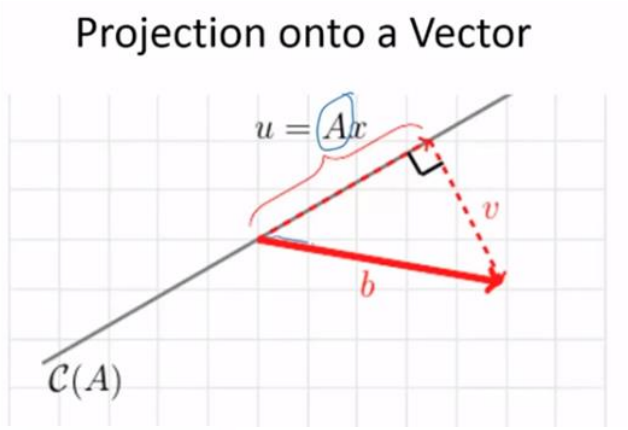
k = 1

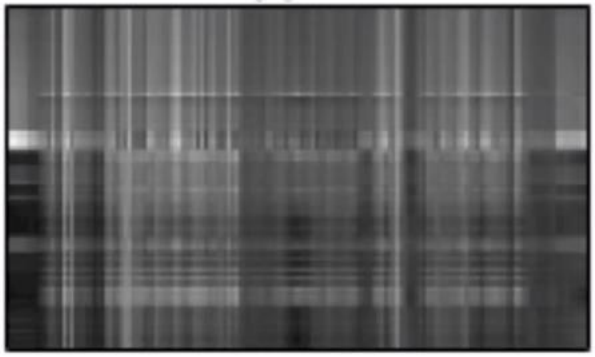
→ 이 그림의 생성 원리 : column vector로 그림을 쪼개고, 각 vector를 한 개의 column vector(배너의 첫번째 column)에 projection 시키게 되면, 즉 이 그림으로 치면 한 개의 vector u 에 다른 b들을 projection시키는 것이다.



- 이번에는 이 배너에서 첫번째랑 가운데의 column을 밑의 matrix A로 해서 column space를 설정하고, 원래 사진의 vector(b)들을 이 column space에 projection했더니, 밑의 사진처럼 조금 선명해진다.

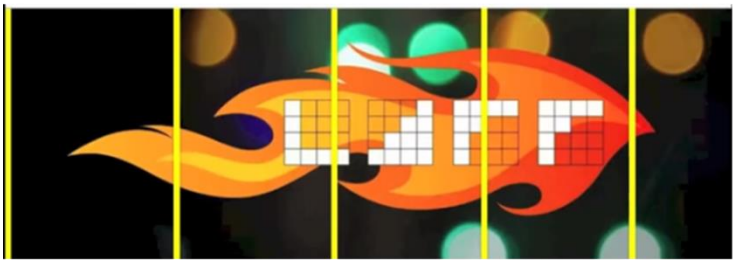


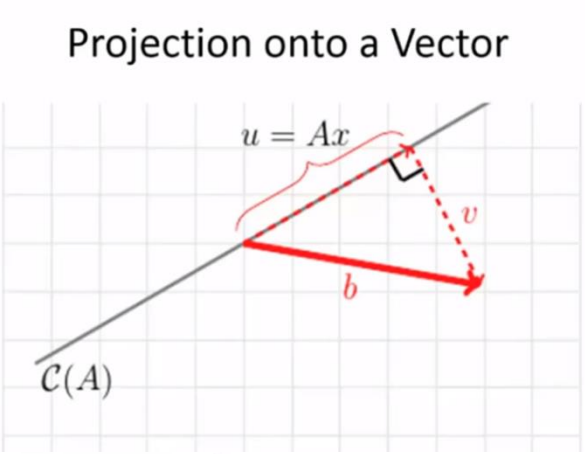




k = 2

- 다시 배너의 5개 column을 행렬 A로 만들어서 이것에 그림의 column vector들을 투사 시키면, 그림이 좀 더 선명해진다.





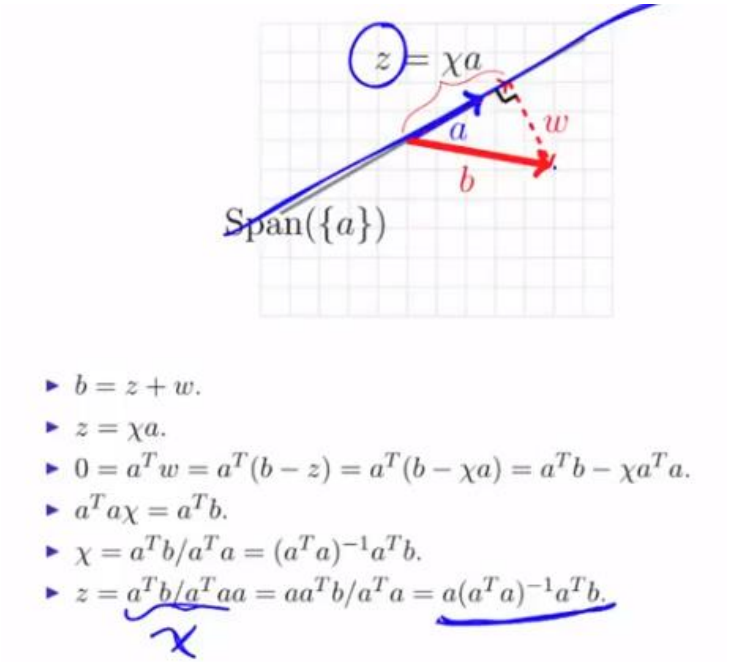
k = 5

- 배너의 column vector를 늘려가면 더욱 더 선명해진다.

k = 10

k = 25

- 수학적 뒷받침 : 아까는 linear least square equation을 했는데, 지금은 이제 data compression에 어떻게 쓰일 수 있는지 알아보자.



- $b = z + w.$
- $z = \chi a.$
- $0 = a^T w = a^T(b - z) = a^T(b - \chi a) = a^T b - \chi a^T a.$
- $a^T a \chi = a^T b.$
- $\chi = a^T b / a^T a = (a^T a)^{-1} a^T b.$
- $z = a^T b / a^T a a = a a^T b / a^T a = a(a^T a)^{-1} a^T b.$

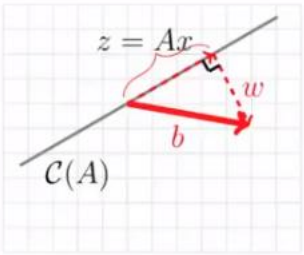
→ $x = (a^T a)^{-1} a^T b$ 여기에 다가

z = x a를 하면 $z = a(a^T a)^{-1} a^T * b$를 할 수 있다. 즉, 어느 b이건 상관없이 a에 projection을 계산할 수 있다. z가 바로 vector b의 projection인 것이다.

- $w = b - z = b - a(a^T a)^{-1} a^T b = (I - a(a^T a)^{-1} a^T) b.$

→ w도 구할 수 있다.

- 스칼라가 아니라 행렬일 때!!



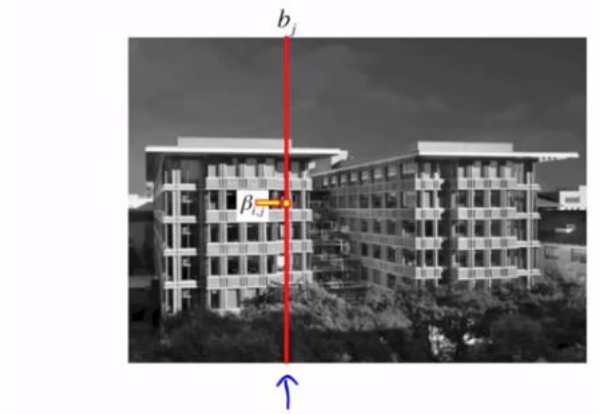
- $b = z + w.$
- $z = Ax.$
- $0 = A^T w = A^T(b - z) = A^T(b - Ax) = A^T b - A^T A x.$
- $A^T A x = A^T b.$
- $x = (A^T A)^{-1} A^T b.$
- $z = A(A^T A)^{-1} A^T b.$
- $w = b - z = b - A(A^T A)^{-1} A^T b = (I - A(A^T A)^{-1} A^T) b.$

- 정리

Given vectors $a, b \in \mathbb{R}^m$

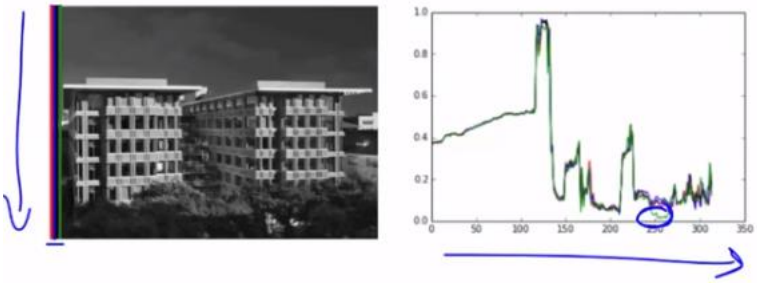
- $a(a^T a)^{-1} a^T$ is the matrix that projects b onto the $\text{Span}(\{a\}) = \mathcal{C}((a))$.
- $I - a(a^T a)^{-1} a^T$ is the matrix that projects b onto the $\text{Span}(\{a\})^\perp = \mathcal{N}((a^T))$.

11.2. Rank-1 approximation



$B \in \mathbb{R}^{m \times n}$: the 2D array of pixels.

$$B = \left(\begin{array}{c|c|c|c} b_0 & b_1 & \cdots & b_{n-1} \end{array} \right)$$



→ 처음 4개의 column의 값들을 plotting하면 비슷하다.

- 그 중 첫번째의 b0을 정해서 a라고 하고, 여기에 다른 b1~bn-1 vector들을 projection 시키자.

- Partition B into columns $B = \left(\begin{array}{c|c|c|c} b_0 & b_1 & \cdots & b_{n-1} \end{array} \right)$.
- Pick $a = b_0$.
- Project b_0 onto $\mathrm{Span}(\{a\})$:

$$a(a^T a)^{-1} a^T b_0 = \underbrace{a(a^T a)^{-1} a^T a}_{\text{Since } b_0 = a} = a.$$

- Project b_1 onto $\mathrm{Span}(\{a\})$:

$$a(a^T a)^{-1} a^T b_1 \approx a$$

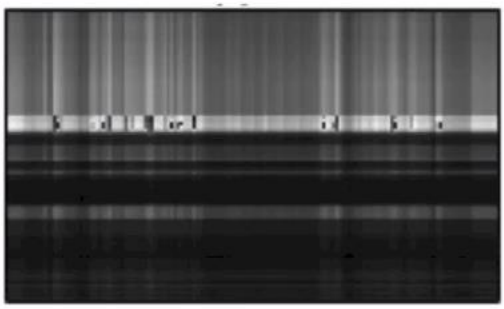
(since b_1 is very close to b_0).

- $\left(\begin{array}{c|c|c|c} a(a^T a)^{-1} a^T b_0 & a(a^T a)^{-1} a^T b_1 & a(a^T a)^{-1} a^T b_2 & \cdots \end{array} \right)$

→ 전체 사진을 a에 projection 시킨 것이 위와 같다.

- $a(a^T a)^{-1} a^T \left(\begin{array}{c|c|c|c} b_0 & b_1 & b_2 & \cdots \end{array} \right) = a(a^T a)^{-1} a^T B.$
- $a(a^T a)^{-1} a^T B = a(\underbrace{(a^T a)^{-1} B^T a}_{y})^T = a y^T$

→ 이 것을 다시 쓰면 $a^T a$는 scalar일 뿐이고 y는 row vector일 뿐이므로 ay^T로 쓰면 된다.



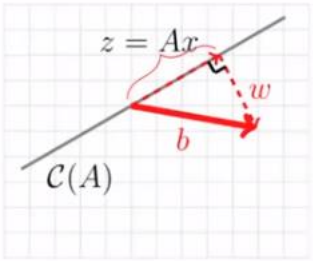
$k = 1$.
Store only $a \in \mathbb{R}^m$ and $y \in \mathbb{R}^n$.
Storing the whole picture: $B \in \mathbb{R}^{m \times n}$.

Given $a \in \mathbb{R}^m$ and $B \in \mathbb{R}^{m \times n}$,

- $B \approx ay^T$ (an outer product), where $y = (a^T a)^{-1} B^T a$.
- Each column of ay^T equals the projection of the corresponding column of B onto $\mathrm{Span}(a)$.
- This is known as a rank-1 approximation of B.
- Not necessarily the *best* rank-1 approximation.

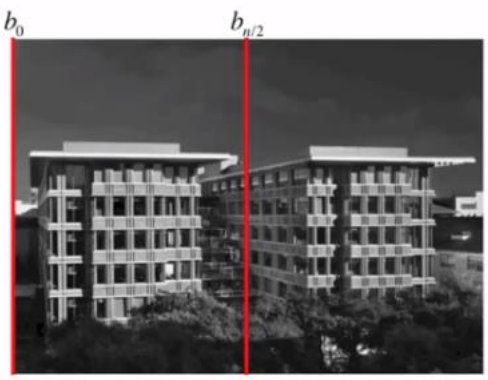
→ B(사진 전체)를 다 쓰지 않고도 a와 y만을 가지고도 이 정보를 저장할 수 있다.

11.3. Rank-2 approximation



- $b = z + w$.
- $z = Ax$.
- $0 = A^T w = A^T(b - z) = A^T(b - Ax) = A^T b - A^T Ax$.
- $A^T Ax = A^T b$. (Normal equation!)
- $x = (A^T A)^{-1} A^T b$.
- $z = A(A^T A)^{-1} A^T b$.
- $w = b - z = b - A(A^T A)^{-1} A^T b = (I - A(A^T A)^{-1} A^T) b$.

- 2개의 column으로 A를 설정해보자!



- Pick $A = \left(a_0 \mid a_1 \right) = \left(b_0 \mid b_{n/2} \right)$.
- $A(A^TA)^{-1}A^Tb_0 = a.$
- $A(A^TA)^{-1}A^Tb_1 \approx a.$
- $\left(A(A^TA)^{-1}A^Tb_0 \mid A(A^TA)^{-1}A^Tb_1 \mid \cdots \right).$
- $A(A^TA)^{-1}A^T \left(b_0 \mid b_1 \mid \cdots \right) = A(A^TA)^{-1}A^TB.$
- $A(A^TA)^{-1}A^TB = AY^T.$

(2×1) 2×m m×n

2×n

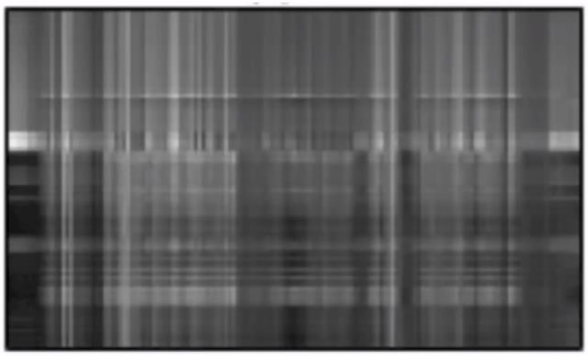
Y^T

$=$ ^{T}

- $AY^T = \left(a_0 \mid a_1 \right)\left(y_0 \mid y_1 \right)^T = \begin{pmatrix} a_0 & a_1 \end{pmatrix}\begin{pmatrix} y_0^T \\ y_1^T \end{pmatrix}$

$= a_0y_0^T + a_1y_1^T.$

→ add two rank-1 matrices(two outer products) = rank-1 update to an outer product



$k = 2.$
Store only $A \in \mathbb{R}^{m\times 2}$ and $Y \in \mathbb{R}^{n\times 2}$.
Storing the whole picture: $B \in \mathbb{R}^{m\times n}$.

- $B \approx AY^T$, where $Y = ((A^TA)^{-1}A^TB)^T = B^TA(A^TA)^{-1}$.
- Each column of AY^T equals the projection of the corresponding column of B onto $\mathcal{C}(A)$.
- This is known as a rank-2 approximation of B.
- Not necessarily the *best* rank-2 approximation.

11.4. Rank-k approximation

$k = 10$

$k = 10$.
Store only $A \in \mathbb{R}^{m \times 10}$ and $Y \in \mathbb{R}^{n \times 10}$.
Storing the whole picture: $B \in \mathbb{R}^{m \times n}$.

→ 10 column씩만 store하면 된다.

- 원리 :

Homework

Let $U \in \mathbb{R}^{m \times k}$ and $V \in \mathbb{R}^{n \times k}$. Then the $m \times n$ matrix UV^T has rank at most k.

True/False

True

- 증명 :

Key: $S, T \subset \mathbb{R}^m$ subspace with $S \subset T$, then $\dim(S) \leq \dim(T)$.
Here $T = \mathcal{C}(\{U\})$ and $S = \mathcal{C}(UV^T)$.

- $\text{rank}(U) = \dim(\mathcal{C}(U)) \leq k$.
- Let $y \in \mathcal{C}(UV^T)$. We will show that then $y \in \mathcal{C}(U)$:

$y \in \mathcal{C}(UV^T)$
$\Rightarrow$ $<$ there exists a $x \in \mathbb{R}^n$ such that $y = UV^T x >$
$y = UV^T x$
$\Rightarrow$ $< z = V^T x >$
$y = Uz$
$\Rightarrow$ $<$ Definition of column space $>$
$y \in \mathcal{C}(U)$.

→ U가 UV^T의 subspace임을 보이면 된다. column space의 정의가 Ax이므로 이를 이용해서 한번만 치환하면 증명이 쉽다.

- low rank approximation을 가장 잘 한 경우가 어떤 경우인가?

- A question: what is the *best* low rank approximation of a matrix?
- Asnwer: The Singular Value Decomposition (SVD)

→ A = UV^T라고 할 때, U의 열 k개가 best, V의 열 k개가 best이면 A도 best가 될 것이다! → SVD

11.5. Orthonormal로 바꾸는 방법 (Gram-Schmidt algorithm)

- 두개의 벡터를 Mutually orthonormal로 바꾸는 방법 (둘 다 직교하면서 length 1으로 맞춤)

Creating orthogonal vectors

Orthogonal (perpendicular) is better!

$$q_0 = x_{n_0}/\|x_{n_0}\|_2 \quad \text{and} \quad q_1 = x_{n_1}^{\perp}/\|x_{n_1}^{\perp}\|_2,$$

where $x_{n_1}^{\perp}$ equals the component of x_{n_1} orthogonal to q_0. We compute this using the Gram-Schmidt algorithm:

- $\rho_{0,0} := \|x_{n_0}\|_2$.
- $q_0 := x_{n_0}/\rho_{0,0}$.
- $\rho_{0,1} := q_0^T x_{n_1}$. $x_{n_1}^{\perp} := x_{n_1} - \rho_{0,1} q_0$.
- $\rho_{1,1} := \|x_{n_1}^{\perp}\|_2$.
- $q_1 := x_{n_1}^{\perp}/\rho_{1,1}$.

→ Gram-Schmidt algorithm.

- mutually orthonormal의 정의 : (1) 각각 길이가 1이어야 하고 (2) dot product가 0이어야 한다.

Let $q_0, q_1, \ldots, q_{k-1} \in \mathbb{R}^m$. Then these vectors are (mutually) orthonormal if for all $0 \le i, j < k$:

$$q_i^T q_j = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{otherwise.} \end{cases}$$

예제1) 이 예는 길이가 1이 아니기 때문에 orthonormal이 아니다. 그러나 루트 2로 각각 나누면 orthonormal이다.

1. The vectors $\begin{pmatrix} -1 \\ 1 \end{pmatrix}, \begin{pmatrix} 1 \\ 1 \end{pmatrix}$ are orthonormal. True/False

False $\begin{pmatrix} -1 \\ 1 \end{pmatrix}/\sqrt{2}, \begin{pmatrix} 1 \\ 1 \end{pmatrix}/\sqrt{2}$

예제2)

1. The vectors $\begin{pmatrix} -\sin(\theta) \\ \cos(\theta) \end{pmatrix}, \begin{pmatrix} \cos(\theta) \\ \sin(\theta) \end{pmatrix}$ are orthonormal.
   True/False

   True
2. The vectors $\begin{pmatrix} \sin(\theta) \\ \cos(\theta) \end{pmatrix}, \begin{pmatrix} \cos(\theta) \\ -\sin(\theta) \end{pmatrix}$ are orthonormal.
   True/False

   True

예제3)

Let $Q \in \mathbb{R}^{m \times k}$ (with $n \le m$) and $Q^T Q = I$. Partition

$$Q = \left(\begin{array}{c|c|c|c} q_0 & q_1 & \cdots & q_{k-1} \end{array} \right).$$

Then $q_0, q_1, \ldots, q_{k-1}$ are mutually orthonormal vectors.

→ 역도 성립한다. mutually orthonomal vector이면(column)이면 $Q^TQ=I$ 이다.

$$
\begin{aligned}
Q^TQ &= \left(\begin{array}{c|c|c|c} q_0 & q_1 & \cdots & q_{k-1} \end{array} \right)^T \left(\begin{array}{c|c|c|c} q_0 & q_1 & \cdots & q_{k-1} \end{array} \right) \\
&= \left(\begin{array}{c} \hline q_0^T \\ \hline q_1^T \\ \hline \vdots \\ \hline q_{k-1}^T \\ \hline \end{array} \right) \left(\begin{array}{c|c|c|c} q_0 & q_1 & \cdots & q_{k-1} \end{array} \right) \\
&= \left(\begin{array}{c|c|c|c} q_0^Tq_0 & q_0^Tq_1 & \cdots & q_0^Tq_{k-1} \\ \hline q_1^Tq_0 & q_1^Tq_1 & \cdots & q_1^Tq_{k-1} \\ \hline \vdots & \vdots & & \vdots \\ \hline q_{k-1}^Tq_0 & q_{k-1}^Tq_1 & \cdots & q_{k-1}^Tq_{k-1} \end{array} \right)
\end{aligned}
$$

예제4) 예제 3에 따라서

1. $\begin{pmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{pmatrix}^T \begin{pmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$
2. $\begin{pmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{pmatrix}^T \begin{pmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$
3. The vectors $\begin{pmatrix} -\sin(\theta) \\ \cos(\theta) \end{pmatrix}, \begin{pmatrix} \cos(\theta) \\ \sin(\theta) \end{pmatrix}$ are orthonormal.

   True/False

   True
4. The vectors $\begin{pmatrix} \sin(\theta) \\ \cos(\theta) \end{pmatrix}, \begin{pmatrix} \cos(\theta) \\ -\sin(\theta) \end{pmatrix}$ are orthonormal.

   True/False

예제5)

Let $q \in \mathbb{R}^m$ be a unit vector (which means it has length one). Then the matrix that projects vectors onto $\mathbf{Span}(\{q\})$ is given by qq^T.

TRUE ▾ ✔ **Answer:** TRUE

The matrix that projects onto $\mathbf{Span}(\{q\})$ is given by $q(q^Tq)^{-1}q^T$. But $q^Tq = 1$ since q is of length one. Thus $q\underbrace{(q^Tq)^{-1}}_{1}q^T = qq^T$.

→ qq^T로 projection할 수 있다.

예제6)

Let $q \in \mathbb{R}^m$ be a unit vector (which means it has length one). Let $x \in \mathbb{R}^m$. Then the component of x in the direction of q (in $\mathbf{Span}(\{q\})$) is given by q^Txq.

FALSE ▾ ✖ **Answer:** TRUE

In the last exercise we saw that the matrix that projects onto $\mathbf{Span}(\{q\})$ is given by qq^T. Thus, the component of x in the direction of q is given by $qq^Tx = q(q^Tx) = q^Txq$ (since q^Tx is a scalar).

→ projection한거를 q^Txq로 표현할 수 있다는 것!!!

예제7) 이제 1개의 벡터가 아니라 여러 개의 벡터, 즉 행렬에 projection 시키는 경우

Let $Q \in \mathbb{R}^{m \times n}$ have orthonormal columns (which means $Q^T Q = I$). Then the matrix that projects vectors onto the column space of Q, $\mathcal{C}(Q)$, is given by QQ^T.

TRUE ▾ ✔ **Answer:** TRUE

The matrix that projects onto $\mathcal{C}(Q)$ is given by $Q(Q^T Q)^{-1} Q^T$. But then
$Q \underbrace{(Q^T Q)^{-1}}_{I^{-1} = I} Q^T = QQ^T$.

예제8)

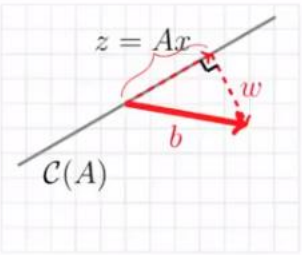
Let $Q \in \mathbb{R}^{m \times n}$ have orthonormal columns (which means $Q^T Q = I$). Then the matrix that projects vectors onto the space orthogonal to the columns of Q, $\mathcal{C}(Q)^{\perp}$, is given by $I - QQ^T$
.

Select an option ▾ **Answer:** TRUE

In the last problem we saw that the matrix that projects onto $\mathcal{C}(Q)$ is given by QQ^T. Hence, the matrix that projects onto the space orthogonal to $\mathcal{C}(Q)$ is given by $I - QQ^T$.

→ 해설

$I - A(A^TA)^{-1}A^T$
$= I - Q\underbrace{(Q^TQ)^{-1}}_{I}Q^T = I - QQ^T$



- $b = z + w$.
- $z = Ax$.
- $0 = A^T w = A^T(b - z) = A^T(b - Ax) = A^T b - A^T Ax$.
- $A^T Ax = A^T b$.
- $x = (A^T A)^{-1} A^T b$.
- $z = A(A^T A)^{-1} A^T b$.
- $w = b - z = b - A(A^T A)^{-1} A^T b = (I - A(A^T A)^{-1} A^T)b$.

11.6. Orthogonal Bases : Gram-Schimdt orthogonalization

- 과정 : linearly independent vector를 set of mutually orthonormal vector로 바꿔준다!
- 목적 :

Given

$$a_0 = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}, a_1 = \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix}, a_2 = \begin{pmatrix} -1 \\ -2 \\ 2 \end{pmatrix}$$

find orthonormal

$$q_0, \quad q_1, \quad \text{and} \quad q_2$$

such that

$$\text{Span}(\{q_0, q_1, q_2\}) = \text{Span}(\{a_0, a_1, a_2\}).$$

- 1단계 : normalize a0

a0를 먼저 손댄다. 방향은 같되 크기를 1로 만든다. → q0로 만든다.

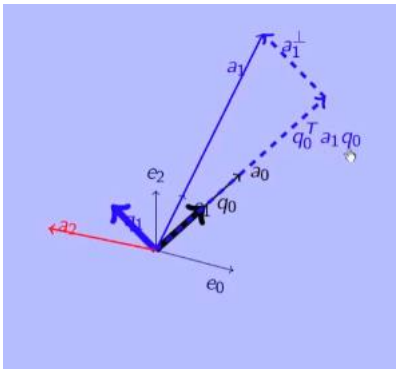
$$q_0 = a_0 / \|a_0\|_2.$$

$$\rho_{0,0} = \left\| \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} \right\|_2 = \sqrt{1^2 + 1^2 + 1^2} = \sqrt{3}$$

$$q_0 = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} / \sqrt{3} = \frac{\sqrt{3}}{3} \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} = \begin{pmatrix} \sqrt{3}/3 \\ \sqrt{3}/3 \\ \sqrt{3}/3 \end{pmatrix}$$

- 2단계 : compute q1

→ a1을 q0와 수직이 되도록 한다. 그리고 크기 1로 normalize한다.



아까 배웠던 기술을 써먹을 수 있다. q0에 projection하려면 q^T x q를 하면 되는 것이다. 그리고 a1에서 그것을 빼고 normalize하면 q1이 된다.

$$a_1^\perp = a_1 - q_0^T a_1 q_0.$$

$$\rho_{0,1} := q_0^T a_1. \qquad \rho_{0,1} = \frac{\sqrt{3}}{3} \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}^T \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix} = 6\frac{\sqrt{3}}{3} = 2\sqrt{3}$$

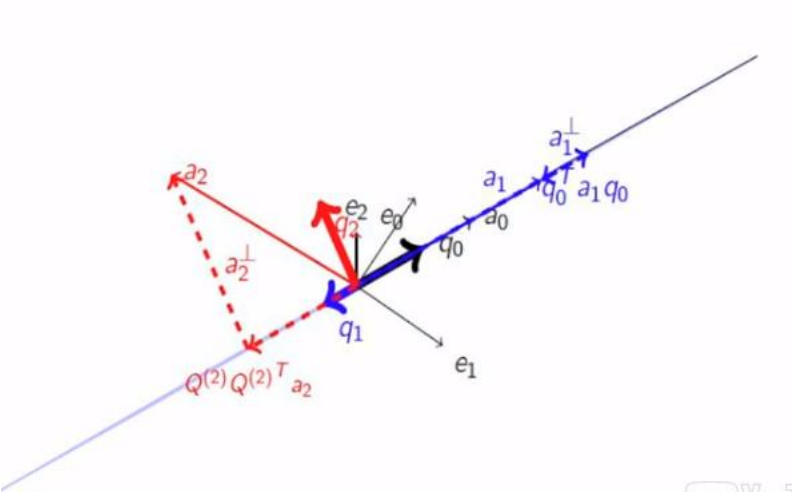
$$a_1^\perp := a_1 - \rho_{0,1} q_0. \qquad a_1^\perp = \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix} - 2\sqrt{3}\frac{\sqrt{3}}{3} \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} = \begin{pmatrix} -1 \\ 0 \\ 1 \end{pmatrix}$$

→ 다음에 크기가 1으로 normalize하기 위해서 q1을 만들기 위해서 다시 길이로 나눈다.

$$\rho_{1,1} := \|a_1^{\perp}\|_2. \qquad \rho_{1,1} = \left\| \begin{pmatrix} -1 \\ 0 \\ 1 \end{pmatrix} \right\|_2 = \sqrt{(-1)^2 + 0^2 + 1^2} = \sqrt{2}$$

$$q_1 := a_1^{\perp}/\rho_{1,1}. \qquad q_1 = \begin{pmatrix} -1 \\ 0 \\ 1 \end{pmatrix} /\sqrt{2} = \frac{\sqrt{2}}{2} \begin{pmatrix} -1 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} -\sqrt{2}/2 \\ 0 \\ \sqrt{2}/2 \end{pmatrix}$$

- 3단계 : q2를 구해야 한다.
아까 q1을 구하는 과정이랑 유사하다.



→ a2를 q0와 q1가 있는 plane에 projection시킨 다음에, a2에서 그 값을 빼면 a2ㅗ가 된다. 그것을 그것의 크기로 나누면 될 것이다.
- q1과 q0는 Q2라는 행렬로 표현될 수 있다.

$$a_2^{\perp} = a_2 - \underbrace{\begin{pmatrix} q_0 & q_1 \end{pmatrix} \begin{pmatrix} q_0 & q_1 \end{pmatrix}^T a_2}_{Q^{(2)}Q^{(2)\,T}a_2}.$$

$QQ^T a_2$를 하면 projection한 벡터가 된다. 따라서 a2에서 그 값을 빼면 수직인 벡터가 나온다.

$$\begin{pmatrix} \rho_{0,2} \\ \rho_{1,2} \end{pmatrix} := \underbrace{\begin{pmatrix} q_0^T a_1 \\ q_1^T a_1 \end{pmatrix}}_{Q^{(2)\,T}a_1}.$$

$$a_2^{\perp} := \underbrace{a_2 - \begin{pmatrix} q_0 & q_1 \end{pmatrix} \begin{pmatrix} \rho_{0,2} \\ \rho_{1,2} \end{pmatrix}}_{a_2 - \rho_{0,2}q_0 - \rho_{1,2}q_0}$$
$$= a_2 - q_0^T a_2 q_0 - q_1^T a_2 q_1$$

→ 이를 두 단계로 나눠서 하면 편하다.

Step 3a: Compute $a_2^\perp$

$$Q^{(2)} = (\, q_0 | q_1 \,) = \left(\frac{\sqrt{3}}{3} \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} \middle| \frac{\sqrt{2}}{2} \begin{pmatrix} -1 \\ 0 \\ 1 \end{pmatrix} \right)$$

$$\begin{pmatrix} \rho_{0,2} \\ \rho_{1,2} \end{pmatrix} := \underbrace{\begin{pmatrix} q_0^T a_1 \\ q_1^T a_1 \end{pmatrix}}_{Q^{(2)^T} a_1}$$

$$\rho_{0,2} = \frac{\sqrt{3}}{3} \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}^T \begin{pmatrix} -1 \\ -2 \\ 2 \end{pmatrix} = \frac{\sqrt{3}}{3}(-1) = -\frac{\sqrt{3}}{3}$$

$$\rho_{1,2} = \frac{\sqrt{2}}{2} \begin{pmatrix} -1 \\ 0 \\ 1 \end{pmatrix}^T \begin{pmatrix} -1 \\ -2 \\ 2 \end{pmatrix} = \frac{\sqrt{2}}{2}(3) = \frac{3\sqrt{2}}{2}$$

$$a_2^\perp := \underbrace{a_2 - (\, q_0 \;\; q_1 \,) \begin{pmatrix} \rho_{0,2} \\ \rho_{1,2} \end{pmatrix}}_{\begin{array}{c} a_2 - \rho_{0,2} q_0 - \rho_{1,2} q_0 \\ = \\ a_2 - q_0^T a_2 q_0 - q_1^T a_2 q_1 \end{array}}$$

$$a_2^\perp = \begin{pmatrix} -1 \\ -2 \\ 2 \end{pmatrix} - \frac{-\sqrt{3}}{3} \frac{\sqrt{3}}{3} \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} - \frac{3\sqrt{2}}{2} \frac{\sqrt{2}}{2} \begin{pmatrix} -1 \\ 0 \\ 1 \end{pmatrix}$$

$$= \begin{pmatrix} -1 \\ -2 \\ 2 \end{pmatrix} - \begin{pmatrix} -1/3 \\ -1/3 \\ -1/3 \end{pmatrix} - \begin{pmatrix} -3/2 \\ 0 \\ 3/2 \end{pmatrix} = \begin{pmatrix} 5/6 \\ -5/3 \\ 5/6 \end{pmatrix}$$

→ 다음에 길이로 나눠서 크기를 1로 만든다.

- 정리

Gram-Schmidt process

- Compute q_0:
  - Normalize a_0:
    - $\rho_{0,0} = \|a_0\|_2$.
    - $q_0 = a_0 / \rho_{0,0}$.
- Compute q_1:
  - Let $a_1^\perp$ equal the component of a_1 orthogonal to q_0:
    - $\rho_{0,1} = q_0^T a_1$.
    - $a_1^\perp = a_1 - \rho_{0,1} q_0$.
  - Normalize $a_1^\perp$:
    - $\rho_{1,1} = \|a_1^\perp\|_2$.
    - $q_1 = a_1^\perp / \rho_{1,1}$.
- Compute q_2:
  - Let $a_2^\perp$ equal the component of a_2 orthogonal to q_0 and q_1:
    - $\rho_{0,2} = q_0^T a_2$.
    - $\rho_{1,2} = q_1^T a_2$.
    - $a_2^\perp = a_2 - \rho_{0,2} q_0 - \rho_{1,2} q_1$.
  - Normalize $a_2^\perp$:
    - $\rho_{2,2} = \|a_2^\perp\|_2$.
    - $q_2 = a_2^\perp / \rho_{2,2}$.
- Compute q_3:
  - ...

11.7. Alternative explanation of Gram-Schmidt process

- 의미 : 나중에 QR factorization과 연관된다.
- 목적 설정 :

Given

$$a_0, a_1, \ldots, a_{n-1} \in \mathbb{R}^m$$

Compute

$$q_0, q_1, \ldots, q_{n-1} \in \mathbb{R}^m$$

such that

$$\mathrm{Span}(\{a_0, a_1, \ldots, a_{n-1}\}) = \mathrm{Span}(\{q_0, q_1, \ldots, q_{n-1}\}).$$

→ 여기까지는 똑같다.

In addition:

$$\begin{aligned}
\mathrm{Span}(\{a_0\}) &= \mathrm{Span}(\{q_0\}) \\
\mathrm{Span}(\{a_0, a_1\}) &= \mathrm{Span}(\{q_0, q_1\}) \\
&\vdots \\
\mathrm{Span}(\{a_0, a_1, \ldots, a_{k-1}\}) &= \mathrm{Span}(\{q_0, q_1, \ldots, q_{k-1}\}) \\
&\vdots \\
\mathrm{Span}(\{a_0, a_1, \ldots, a_{n-1}\}) &= \mathrm{Span}(\{q_0, q_1, \ldots, q_{n-1}\})
\end{aligned}$$

→ 여기서부터는 좀 다르다.

- 먼저 q0를 먼저 구해보자

$$\mathrm{Span}(\{a_0\}) = \mathrm{Span}(\{q_0\})$$

$$a_0 = \rho_{0,0} q_0$$

$$\begin{aligned}
\rho_{0,0} &:= \|a_0\|_2 \\
q_0 &:= a_0 / \rho_{0,0}.
\end{aligned}$$

- q1을 구해보자.

Computing q_1

- $\mathrm{Span}(\{a_0, a_1\}) = \mathrm{Span}(\{q_0, q_1\})$
- $a_1 = \rho_{0,1} q_0 + \rho_{1,1} q_1$
- $q_0^T q_1 = 0$ and $q_1^T q_1 = 1$
- $$\begin{aligned}
q_0^T a_1 &= q_0^T (\rho_{0,1} q_0 + \rho_{1,1} q_1) \\
&= q_0^T \rho_{0,1} q_0 + q_0^T \rho_{1,1} q_1 \\
&= \rho_{0,1} \underbrace{q_0^T q_0}_{1} + \rho_{1,1} \underbrace{q_0^T q_1}_{0} \\
&= \rho_{0,1}
\end{aligned}$$
- $\underbrace{\rho_{1,1} q_1}_{a_1^\perp} = a_1 - \underbrace{q_0^T a_1}_{\rho_{0,1}} q_0$

$$\begin{aligned}
\rho_{0,1} &:= q_0^T a_1 \\
a_1^\perp &:= a_1 - \rho_{0,1} q_0 \\
\rho_{1,1} &:= \|a_1^\perp\|_2 \\
q_1 &:= a_1^\perp / \rho_{1,1}.
\end{aligned}$$

→ q0, q1에 span하기 때문에 a1을 두 개의 linear combination으로 표현할 수 있어야 한다.
→ 목적은 q1을 구하는 것이다. 저 식에서 ρ0,1은 구했다. q1을 구하기 위해서 나머지를 우측으로 이항하고 마지막으로 ρ1,1(길이)로 나누면 된다.

- q2를 구해보자.

Computing q_2

- $a_2 = \rho_{0,2}q_0 + \rho_{1,2}q_1 + \rho_{2,2}q_2$
- $$\begin{aligned} q_0^T a_2 &= q_0^T(\rho_{0,2}q_0 + \rho_{1,2}q_1 + \rho_{2,2}q_2) \\ &= q_0^T\rho_{0,2}q_0 + q_0^T\rho_{1,2}q_1 + q_0^T\rho_{2,2}q_2 \\ &= \rho_{0,2}\underbrace{q_0^Tq_0}_{1} + \rho_{1,2}\underbrace{q_0^Tq_1}_{0} + \rho_{2,2}\underbrace{q_0^Tq_2}_{0} = \rho_{0,2} \end{aligned}$$
- $$\begin{aligned} q_1^T a_2 &= q_1^T(\rho_{0,2}q_0 + \rho_{1,2}q_1 + \rho_{2,2}q_2) \\ &= q_1^T\rho_{0,2}q_0 + q_1^T\rho_{1,2}q_1 + q_1^T\rho_{2,2}q_2 \\ &= \rho_{1,2}\underbrace{q_1^Tq_0}_{0} + \rho_{1,2}\underbrace{q_1^Tq_1}_{1} + \rho_{2,2}\underbrace{q_1^Tq_2}_{0} = \rho_{1,2} \end{aligned}$$
- $\rho_{2,2}q_2 = a_2^\perp = a_2 - \rho_{0,2}q_0 - \rho_{1,2}q_1$
- $\rho_{2,2} = \|a_2^\perp\|_2$
- $q_2 = a_2^\perp/\rho_{2,2}$

→ a2를 q0, q1, q2의 linear combination으로 표현할 수 있다. 다음 양변에 아까처럼 $q0^T$를 곱하면 ρ0,2만이 남는다. ρ1,2도 마찬가지로 구할 수 있다.

$$\begin{aligned} \rho_{0,2} &:= q_0^T a_2 \\ \rho_{1,2} &:= q_1^T a_2 \\ a_2^\perp &:= a_2 - \rho_{0,2}q_0 - \rho_{1,2}q_1 \\ \rho_{2,2} &:= \|a_2^\perp\|_2 \\ q_2 &:= a_2^\perp/\rho_{2,2}. \end{aligned}$$

→ 이와 같은 단계로 진행하면 된다.

- qk를 계산하려면

- $a_k = \rho_{0,k}q_0 + \rho_{1,2}q_1 + \ldots + \rho_{k-1,k}q_{k-1} + \rho_{k,k}q_k$
- $i < k$:
  $$\begin{aligned} q_i^T a_k &= q_i^T(\rho_{0,k}q_0 + \rho_{1,2}q_1 + \ldots + \rho_{k-1,k}q_{k-1} + \rho_{k,k}q_k) \\ &= \rho_{i,k} \end{aligned}$$
- $\rho_{k,k}q_k = a_k^\perp = a_k - \rho_{0,k}q_0 - \rho_{1,k}q_1 - \cdots - \rho_{k-1,k}q_{k-1}$
- $\rho_{k,k} = \|a_k^\perp\|_2$
- $q_k = a_k^\perp/\rho_{k,k}$

$$\rho_{0,k} := q_0^T a_k$$
$$\rho_{1,k} := q_1^T a_k$$
$$\vdots$$
$$\rho_{k-1,k} := q_{k-1}^T a_k$$

$$a_k^\perp := a_k - \rho_{0,k} q_0 - \rho_{1,k} q_1 - \cdots - \rho_{k-1,k} q_{k-1}$$

$$\rho_{k,k} := \|a_k^\perp\|_2$$

$$q_k := a_k^\perp / \rho_{k,k}.$$

→ 이를 한번에 matrix-vector multiplication을 하면 한방에 구할 수 있고, 빼는 것도 한방에 뺄 수 있다. 이것이 바로 matrix의 힘!!

Computing q_k

$$\left.\begin{array}{l} \rho_{0,k} := q_0^T a_k \\ \rho_{1,k} := q_1^T a_k \\ \vdots \\ \rho_{k-1,k} := q_{k-1}^T a_k \end{array}\right\} \begin{pmatrix} \rho_{0,k} \\ \rho_{1,k} \\ \vdots \\ \rho_{k-1,k} \end{pmatrix} := \begin{pmatrix} q_0^T \\ q_1^T \\ \vdots \\ q_{k-1}^T \end{pmatrix} a_k = \left(q_0 \middle| \cdots \middle| q_{k-1} \right)^T a_k$$

$$\begin{aligned} a_k^\perp &:= a_k - \rho_{0,k} q_0 - \rho_{1,k} q_1 - \cdots - \rho_{k-1,k} q_{k-1} \\ &= a_k - \left(q_0 \middle| q_1 \middle| \cdots \middle| q_{k-1} \right) \begin{pmatrix} \rho_{0,k} \\ \rho_{1,k} \\ \vdots \\ \rho_{k-1,k} \end{pmatrix} \end{aligned}$$

$$\rho_{k,k} := \|a_k^\perp\|_2$$

$$q_k := a_k^\perp / \rho_{k,k}.$$

11.8. QR factorization

- 역사 : LU factorization (week 6) → LU factorization with partial pivoting (week 7) → Cholesky factorization (week8) ➔ 이제는!!! QR factorization

- Gram-Schimdt process : compute q0, q1, q2
- 행렬로 조진다.

$$\begin{pmatrix} a_0 \middle| a_1 \middle| & \middle| & \end{pmatrix}$$
$$= \begin{pmatrix} \rho_{0,0} q_0 \middle| \rho_{0,1} q_0 + \rho_{1,1} q_1 \middle| & \middle| & \end{pmatrix}$$
$$= \underbrace{\begin{pmatrix} q_0 \middle| q_1 \middle| & \middle| & \end{pmatrix}}_{Q} \underbrace{\begin{pmatrix} \rho_{0,0} & \rho_{0,1} & & \\ 0 & \rho_{1,1} & & \\ & & & \\ & & & \end{pmatrix}}_{R}$$

Summary

- The Gram-Schmidt process computes

$$A = \underbrace{\left(a_0 \mid a_1 \mid \cdots \mid a_{n-1} \right)}_{A}$$

$$= \underbrace{\left(q_0 \mid q_1 \mid \cdots \mid q_{n-1} \right)}_{Q} \underbrace{\left(\begin{array}{c|c|c|c} \rho_{0,0} & \rho_{0,1} & \cdots & \rho_{0,n-1} \\ \hline 0 & \rho_{1,1} & \cdots & \rho_{1,n-1} \\ \hline \vdots & \vdots & \ddots & \vdots \\ \hline 0 & 0 & \cdots & \rho_{n-1,n-1} \end{array} \right)}_{R}$$

$$= QR.$$

→ 와!! A = QR 로 분리를 했다. 단, R의 column vector는 linearly independent하다.

- QR 본격 시동

$Ax \approx b$ "best" solution: $x = (A^T A)^{-1} A^T b$

$A = QR$ where $Q^T Q = I$

$$x = (A^T A)^{-1} A^T b = ((\underbrace{QR}_{A})^T (\underbrace{QR}_{A}))^{-1} (\underbrace{QR}_{A})^T b$$

$$= (R^T \underbrace{Q^T Q}_{I} R)^{-1} R^T Q^T b = (R^T R)^{-1} R^T Q^T b$$

$$= R^{-1} \underbrace{R^{-T} R^T}_{I} Q^T b = R^{-1} Q^T b.$$

$Ax \approx b$ "best" solution solves: $R\hat{x} = Q^T b$

→ Ax=b에서 Az=b hat이 된다. 이 때 A=QR을 대입하고 $(AB)^T=B^TA^T$와 $(AB)^{-1}=B^{-1}A^{-1}$라는 것을 활용하면 간단하게 정리된다. 이 때, $R^{-1}Q^T$를 직접 계산할 수는 있지만 또 역행렬이 있으므로 양변에 R을 곱하게 되면 $Rx=Q^Tb$를 풀면 된다. R은 upper triangular matrix이다.

$$Ax \approx b$$

- Factor: $A = QR$ where $Q^T Q = I$ and R is upper triangular.
- Solve $Rx = Q^T b$.
- Columns of A must be linearly independent.

→ A의 역행렬이 존재해야 하므로 column이 서로 linearly independent해야 한다.

11.8.2. FLAME notation으로 어떻게 하는가.

$$\underbrace{\left(\begin{array}{c|c|c} A_0 & a_1 & A_2 \end{array}\right)}_{A} = \underbrace{\left(\begin{array}{c|c|c} Q_0 & q_1 & Q_2 \end{array}\right)}_{Q} \underbrace{\left(\begin{array}{c|c|c} R_{00} & r_{01} & R_{02} \\ \hline 0 & \rho 11 & r_{12}^T \\ \hline 0 & 0 & R_{22} \end{array}\right)}_{R}$$

so that

$$\left(\begin{array}{c|c|c} A_0 & a_1 & A_2 \end{array}\right) =$$

$$\left(\begin{array}{c|c|c} Q_0 R_{00} & Q_0 r_{01} + \rho_{11} q_1 & Q_0 R_{02} + q_0 r_{12}^T + Q_2 R_{22} \end{array}\right).$$

→ 첫번째 항은 Gram-Schmidt process로 정리가 된다. 그런데 2번째 항부터 조금 복잡해진다. 두 번째 식 a1 = ~ 양변에 Q^T를 곱해주면, Q^TQ는 I가 되고 Q^Tq_1은 0이 된다. 따라서 r 01만 남게 된다.

$$\begin{aligned} Q_0^T a_1 &= Q_0^T (Q_0 r_{01} + \rho_{11} q_1) \\ &= Q_0^T Q_0 r_{01} + \rho_{11} Q_0^T q_1 \\ &= r_{01} \end{aligned}$$

→ 다음 a1과 Q0r01을 알기 때문에 q1까지 구할 수 있다.

$$\left(\begin{array}{c|c|c} A_0 & a_1 & A_2 \end{array}\right) =$$

$$\left(\begin{array}{c|c|c} Q_0 R_{00} & Q_0 r_{01} + \rho_{11} q_1 & Q_0 R_{02} + q_0 r_{12}^T + Q_2 R_{22} \end{array}\right)$$

$$a_1 = Q_0 r_{01} + \rho_{11} q_1$$

$$\begin{aligned} a_1^\perp &= a_1 - Q_0 r_{01} \\ \rho_{11} &= \|a_1^\perp\|_2 \\ q_1 &= a_1^\perp / \rho_{11} \end{aligned}$$

Algorithm: $[Q, R] := \mathrm{QR}(A, Q, R)$

Partition $A \rightarrow \left(\begin{array}{c|c} A_L & A_R \end{array}\right)$, $Q \rightarrow \left(\begin{array}{c|c} Q_L & Q_R \end{array}\right)$, $R \rightarrow \left(\begin{array}{c|c} R_{TL} & R_{TR} \\ \hline R_{BL} & R_{BR} \end{array}\right)$

where A_L and Q_L have 0 columns, R_{TL} is 0×0

while $n(A_L) < n(A)$ **do**

Repartition

$$\left(\begin{array}{c|c} A_L & A_R \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} A_0 & a_1 & A_2 \end{array}\right), \left(\begin{array}{c|c} Q_L & Q_R \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} Q_0 & q_1 & Q_2 \end{array}\right),$$

$$\left(\begin{array}{c|c} R_{TL} & R_{TR} \\ \hline R_{BL} & R_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} R_{00} & r_{01} & R_{02} \\ \hline r_{10}^T & \rho_{11} & r_{12}^T \\ \hline R_{20} & r_{21} & R_{22} \end{array}\right)$$

$r_{01} := Q_0^T a_1$

$a_1^\perp := a_1 - Q_0 r_{01}$

$\rho_{11} := \|a_1^\perp\|_2$

$q_1 = a_1^\perp / \rho_{11}$

Continue with

$$\left(\begin{array}{c|c} A_L & A_R \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} A_0 & a_1 & A_2 \end{array}\right), \left(\begin{array}{c|c} Q_L & Q_R \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} Q_0 & q_1 & Q_2 \end{array}\right),$$

$$\left(\begin{array}{c|c} R_{TL} & R_{TR} \\ \hline R_{BL} & R_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} R_{00} & r_{01} & R_{02} \\ \hline r_{10}^T & \rho_{11} & r_{12}^T \\ \hline R_{20} & r_{21} & R_{22} \end{array}\right)$$

endwhile

11.9. Change of basis

- 미리 학습해야 할 mutually orthonormal column의 행렬 성질

If $Q \in \mathbb{R}^{n \times n}$ has mutually orthonormal columns then which of the following are true:

1. $Q^T Q = I$ True/False
2. $QQ^T = I$ True/False
3. $QQ^{-1} = I$ True/False
4. $Q^{-1} = Q^T$ True/False

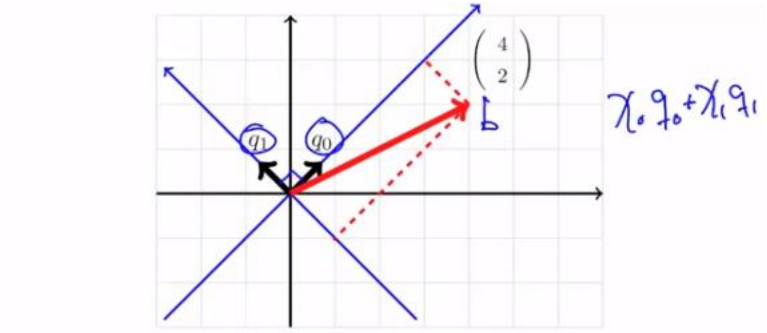
→ 역행렬의 성질을 이용하면 쉽게 증명을 할 수 있다. 여기서 가장 중요한 것은 $QQ^T=I$이다.

- 목표 : q0, q1이 각각

The vectors

$$q_0 = \frac{\sqrt{2}}{2}\begin{pmatrix} 1 \\ 1 \end{pmatrix} = \begin{pmatrix} \frac{\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} \end{pmatrix}, \quad q_1 = \frac{\sqrt{2}}{2}\begin{pmatrix} -1 \\ 1 \end{pmatrix} = \begin{pmatrix} -\frac{\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} \end{pmatrix}$$

로 주어졌을 때, b를 q0, q1의 linear combination으로 표현하고 싶다.



- $\left[\frac{\sqrt{2}}{2}\begin{pmatrix} 1 \\ 1 \end{pmatrix}\right]^T \begin{pmatrix} 4 \\ 2 \end{pmatrix} =$

(1) 일단, b를 q0에 projection시켜보자. 그럴려면, $q0^T$ * b * q0를 하면 된다.
(2) b를 q1에 projection시켜보자. 그럴려면, $q1^T$ * b * q1을 하면 된다.
(1)과 (2)를 더하면 b를 표현할 수 있다.

- $\left[\frac{\sqrt{2}}{2}\begin{pmatrix} 1 \\ 1 \end{pmatrix}\right]^T \begin{pmatrix} 4 \\ 2 \end{pmatrix} = 6\frac{\sqrt{2}}{2} = 3\sqrt{2}$
- $\left[\frac{\sqrt{2}}{2}\begin{pmatrix} -1 \\ 1 \end{pmatrix}\right]^T \begin{pmatrix} 4 \\ 2 \end{pmatrix} = (-2)\frac{\sqrt{2}}{2} = -\sqrt{2}$
- $3\sqrt{2}q_0 - \sqrt{2}q_1$

- 임의의 벡터 b에 대해서 생각해보자. unit basis vector말고 a0, a1, a2, .. an-1로 표현하고 싶다면 어떻게 하면 될까? 똑같이 하면 된다.

$$\begin{aligned} b &= \begin{pmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_{n-1} \end{pmatrix} \\ &= \beta_0 e_0 + \beta_1 e_1 + \cdots + \beta_{n-1} e_{n-1} \\ &= ?\, a_0 + ?\, a_1 + \cdots + ?\, a_{n-1} \\ &= \chi_0 a_0 + \chi_1 a_1 + \cdots + \chi_{n-1} a_{n-1}. \end{aligned}$$

→ A가 역행렬이 있으므로 아래와 같이 쓸 수 있다. a0 ~ an-1 자체가 basis이므로 서로 linearly independent하기 때문이다. Ax는 다시 linear combination으로 나타낼 수 있다. 즉, b = Ax이다.

$$\begin{aligned} b &= \underbrace{AA^{-1}}_{I} b \\ &= Ax = \chi_0 a_0 + \chi_1 a_1 + \cdots + \chi_{n-1} a_{n-1}. \end{aligned}$$

- b를 orthonormal bases로 표현하고 싶을 때는 어떻게 되는가?

$$
\begin{aligned}
b &= \begin{pmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_{n-1} \end{pmatrix} \\
&= \beta_0 e_0 + \beta_1 e_1 + \cdots + \beta_{n-1} e_{n-1} \\
&= ?\, q_0 + ?\, q_1 + \cdots + ?\, q_{n-1} \\
&= \chi_0 q_0 + \chi_1 q_1 + \cdots + \chi_{n-1} q_{n-1}.
\end{aligned}
$$

$$
\begin{aligned}
b &= \underbrace{QQ^{-1}}_{I} b = QQ^T b \\
&= \left(\begin{array}{c|c|c|c} q_0 & q_1 & \cdots & q_{n-1} \end{array} \right) \left(\begin{array}{c|c|c|c} q_0 & q_1 & \cdots & q_{n-1} \end{array} \right)^T b \\
&= \left(\begin{array}{c|c|c|c} q_0 & q_1 & \cdots & q_{n-1} \end{array} \right) \begin{pmatrix} \frac{q_0^T}{} \\ \hline \frac{q_1^T}{} \\ \hline \vdots \\ \hline \frac{q_{n-1}^T}{} \end{pmatrix} b \\
&= \left(\begin{array}{c|c|c|c} q_0 & q_1 & \cdots & q_{n-1} \end{array} \right) \begin{pmatrix} \frac{q_0^T b}{} \\ \hline \frac{q_1^T b}{} \\ \hline \vdots \\ \hline \frac{q_{n-1}^T b}{} \end{pmatrix} \\
&= q_0^T b\, q_0 + q_1^T b\, q_1 + \cdots + q_{n-1}^T b\, q_{n-1}.
\end{aligned}
$$

→ QQ-1=QQT를 사용해서 다시 표현한 것이다. 따라서 이와 같이 q0, q1, q2로 표현이 가능하다.

11.10. The Best Low rank approximation : SVD (Single value decomposition)

- Single value decomposition의 기능 : best low rank approximation을 준다.

Let $B \in \mathbb{R}^{m \times n}$. Then

$$B = U \Sigma V^T$$

where

- $U \in \mathbb{R}^{m \times r}$ and $U^T U = I$.
- $\Sigma \in \mathbb{R}^{r \times r}$ is a diagonal matrix with positive diagonal elements that are ordered so that $\sigma_0 \geq \sigma_1 \geq \cdots \geq \sigma_{r-1} > 0$.
- $V \in \mathbb{R}^{n \times r}$ and $V^T V = I$.
- r equals the rank of matrix B.

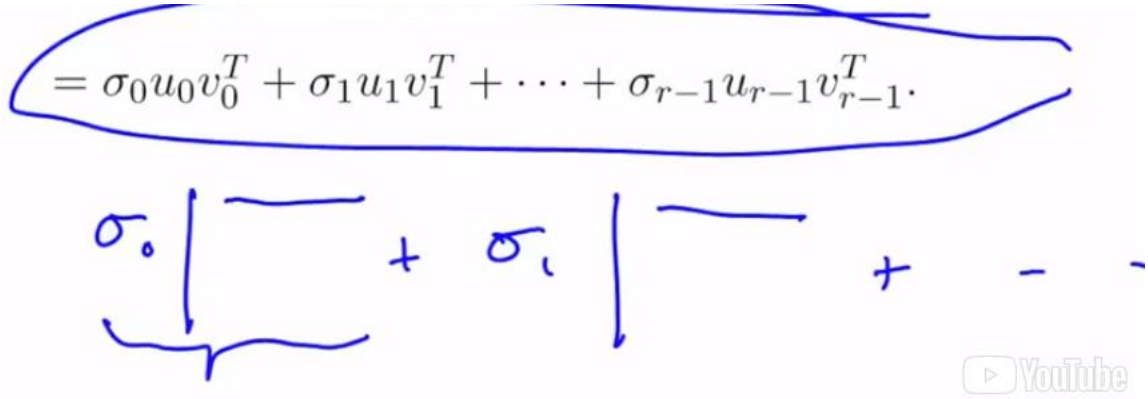
→ U와 V는 orthonormal matrix이며 r은 rank이며 행렬 B의 linearly independent column 개수이다.

→ 시그마는 diagonal하며 singular value인 것이 큰 것부터 차례대로 나열되어 있는 행렬이다.

$$B = \underbrace{\begin{pmatrix} u_0 & u_1 & \cdots & u_{r-1} \end{pmatrix}}_{U} \underbrace{\begin{pmatrix} \sigma_0 & 0 & \cdots & 0 \\ 0 & \sigma_1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \sigma_{r-1} \end{pmatrix}}_{\Sigma} \underbrace{\begin{pmatrix} v_0 & v_1 & \cdots & v_{r-1} \end{pmatrix}^T}_{V}$$

$$= \sigma_0 u_0 v_0^T + \sigma_1 u_1 v_1^T + \cdots + \sigma_{r-1} u_{r-1} v_{r-1}^T.$$

→ 옛날에 나왔듯이, diagonal matrix를 곱하게 되면 이것은 스칼라 곱이 된 다는 것을 배웠다. 따라서 앞 뒤로 곱해진 형태가 된다. 아래를 보면 rank-1 matrices가 더해진 형태라는 것을 알 수 있다.



- 다시 말하면

$$B = \underbrace{\begin{pmatrix} u_0 & u_1 & \cdots & u_{k-1} & u_k & \cdots & u_{r-1} \end{pmatrix}}_{U} \underbrace{\left(\begin{array}{cccc|ccc} \sigma_0 & 0 & \cdots & 0 & 0 & \cdots & 0 \\ 0 & \sigma_1 & \cdots & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & \sigma_{k-} & 0 & \cdots & 0 \\ \hline 0 & 0 & \cdots & 0 & \sigma_k & \cdots & 0 \\ \vdots & \vdots & & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & 0 & \cdots & \sigma_{r-1} \end{array}\right)}_{\Sigma} \underbrace{\begin{pmatrix} v_0 & v_1 & \cdots & v_{k-1} & v_k & \cdots & v_{r-1} \end{pmatrix}^T}_{V}$$

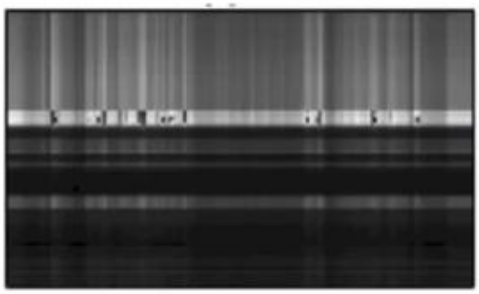
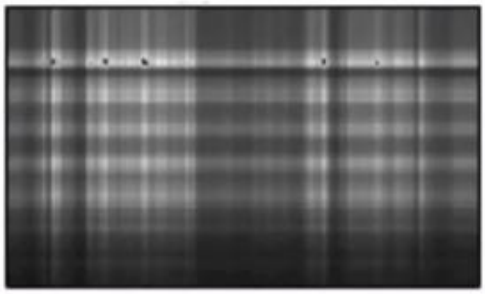
$$= \sigma_0 u_0 v_0^T + \sigma_1 u_1 v_1^T + \cdots + \sigma_{k-1} u_{k-1} v_{k-1}^T + \sigma_k u_k v_k^T + \cdots + \sigma_{r-1} u_{r-1} v_{r-1}^T.$$

→ Σ matrix에서 k 번째까지의 singular value를 선택하고 나머지 항을 0으로 두면 best rank-k approximation이 된다.

$$B \approx \underbrace{\begin{pmatrix} u_0 & u_1 & \cdots & u_{k-1} \end{pmatrix}}_{U} \underbrace{\begin{pmatrix} \sigma_0 & 0 & \cdots & 0 \\ 0 & \sigma_1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \sigma_{k-1} \end{pmatrix}}_{\Sigma} \underbrace{\begin{pmatrix} v_0 & v_1 & \cdots & v_{k-1} \end{pmatrix}^T}_{V}$$

$$= \sigma_0 u_0 v_0^T + \sigma_1 u_1 v_1^T + \cdots + \sigma_{k-1} u_{k-1} v_{k-1}^T.$$

- 품질 차이를 경험해보시라! crude way vs singular value decomposition

$A(A^TA)^{-1}A^TB$	$U_L\Sigma_{TL}V_L^T$
	
$k = 1$	$k = 1$
$A(A^TA)^{-1}A^TB$	$U_L\Sigma_{TL}V_L^T$
	
$k = 50$	$k = 50$

- linear least square를 svd를 이용해서 푸는 방법

Solving the linear least-squares problem with the SVD

- $A = U\Sigma V^T$ where $U \in \mathbb{R}^{m\times r}$, $\Sigma \in \mathbb{R}^{r\times r}$, $V \in \mathbb{R}^{n\times r}$.
- Assume A has rank n (i.e., $r = n$)
- Solve $Ax \approx b$:

$$
\begin{aligned}
x &= (A^TA)^{-1}A^Tb \\
&= ((U\Sigma V^T)^T(U\Sigma V^T))^{-1}(U\Sigma V^T)^Tb \\
&= (V\Sigma^TU^TU\Sigma V^T)^{-1}V\Sigma^TU^Tb \\
&= (V\Sigma\Sigma V^T)^{-1}V\Sigma U^Tb \\
&= ((V^T)^{-1}(\Sigma\Sigma)^{-1}V^{-1})V\Sigma U^Tb \\
&= V^T\Sigma^{-1}\Sigma^{-1}\Sigma U^Tb \\
&= V^T\Sigma^{-1}U^Tb
\end{aligned}
$$

12장. Eigenvalues and Eigenvectors : diagonalizing a matrix

12.2.1. The Algebraic Eigenvalue Problem
12.2.2. Simple Examples
12.2.3. Diagonalizing
12.2.4. Eigenvalues and Eigenvectors of 33 Matrices
The General Case
12.3.1. Eigenvalues and Eigenvectors of nn matrices: Special Cases
12.3.2. Eigenvalues of nn Matrices
12.3.3. Diagonalizing, Again
12.3.4. Properties of Eigenvalues and Eigenvectors
Practical Methods for Computing Eigenvectors and Eigenvalues
12.4.1. Predicting the Weather, One Last Time
12.4.2. The Power Method
12.4.3. In Preparation for the Enrichment
Enrichment
12.5.1. The Inverse Power Method
12.5.2. The Rayleigh Quotient Iteration
12.5.3. More Advanced Technique

12.1. Algebraic Eigenvalue problem

- Ax =λx
- 이게 성립하려면 무조건 square matrix여야 한다. λ가 scalar이기 때문이다.
- eigenvalue λ는 한 개의 값은 아니다. infinite하다.

- Lambda를 구해보자!

- $Ax = \lambda x, \quad x \neq 0$
- $Ax - \lambda x = 0, \quad x \neq 0$
- $Ax - \lambda I x = 0, \quad x \neq 0$
- $(A - \lambda I)x = 0, \quad x \neq 0$
- Find λ so that $A - \lambda I$ is singular.
- Find (all) $x \neq 0$ so that $(A - \lambda I)x = 0$.
- Find (all) $x \in \mathcal{N}(A - \lambda I)$ (discard $x = 0$).

→ eigenvector를 표현하는데 가장 쉬운 방법은 null space로 표현하는 방법이다. (A-λI)x = 0으로!

12.1.1. Diagonal matrix

Find all (λ, x) (eigenpairs):

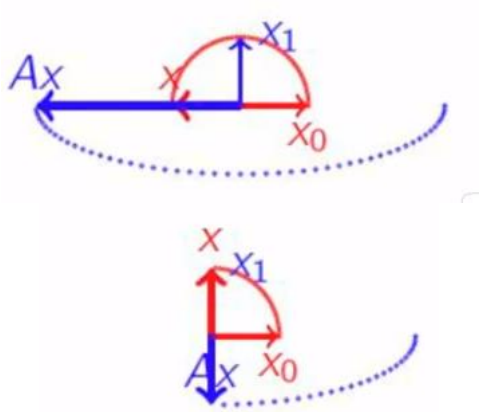
$$\begin{pmatrix} 3 & 0 \\ 0 & -1 \end{pmatrix} x = \lambda x$$

$$\left[\begin{pmatrix} 3 & 0 \\ 0 & -1 \end{pmatrix} - \lambda I \right] x = 0$$

$$\begin{pmatrix} 3-\lambda & 0 \\ 0 & -1-\lambda \end{pmatrix} x = 0$$

$\lambda = 3$ $\lambda = -1$

→ determinant를 통해서 lambda를 구한다.



Ax = 3x, Ax = -x를 각각 표현하는 그림이다.

- diagonal matrix의 eigen vector

Consider the diagonal matrix $\begin{pmatrix} \delta_0 & 0 & \cdots & 0 \\ 0 & \delta_1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \delta_{n-1} \end{pmatrix}$.

Eigenpairs for this matrix are given by $(\delta_0, e_0), (\delta_1, e_1), \cdots, (\delta_{n-1}, e_{n-1})$, where e_j equals the jth unit basis vector.

Always ✔

- n x n matrix의 대각행렬에서 a i,i eigenvalue는 i 번째 eigenvector랑 관련이 있다.

$$A = \begin{pmatrix} \alpha_{0,0} & 0 & 0 & \cdots & 0 \\ 0 & \alpha_{1,1} & 0 & \cdots & 0 \\ 0 & 0 & \alpha_{2,2} & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & \alpha_{n-1,n-1} \end{pmatrix}.$$

e_i is an eigenvector associated with eigenvalue $\alpha_{i,i}$.

Always/Sometimes/Never

Answer:

$$\left(\begin{array}{c|c|c} A_{00} & 0 & 0 \\ \hline 0 & \alpha_{11} & 0 \\ \hline 0 & 0 & A_{22} \end{array}\right) \left(\begin{array}{c} 0 \\ \hline 1 \\ \hline 0 \end{array}\right) = \left(\begin{array}{c} 0 \\ \hline \alpha_{11} \\ \hline 0 \end{array}\right) = \alpha_{11} \left(\begin{array}{c} 0 \\ \hline 1 \\ \hline 0 \end{array}\right)$$

12.1.2. Upper triangular matrix

Homework

Find all (λ, x) (eigenpairs):

$$\begin{pmatrix} 3 & 1 \\ 0 & -1 \end{pmatrix} x = \lambda x$$

$$\left[\begin{pmatrix} 3 & 1 \\ 0 & -1 \end{pmatrix} - \lambda I\right] x = 0$$

$$\begin{pmatrix} 3-\lambda & 1 \\ 0 & -1-\lambda \end{pmatrix} x = 0$$

$\lambda = 3$ $\lambda = -1$

$\begin{pmatrix} 0 & 1 \\ 0 & -4 \end{pmatrix} \begin{pmatrix} x_0 \\ x_1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$

$x_0 = 1$

$\begin{pmatrix} 0 & 1 \\ 0 & -4 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$

$\Rightarrow \begin{pmatrix} 4 & 1 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} x_0 \\ x_1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$

$\begin{pmatrix} 4 & 1 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} -1/4 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$

- n x n upper triangular matrix에서

Let $A = \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline 0 & \alpha_{11} & a_{12}^T \\ \hline 0 & 0 & A_{22} \end{array}\right)$, where A_{00} is square. Then α_{11} is an eigenvalue of A and $\begin{pmatrix} -(A_{00} - \alpha_{11} I)^{-1} a_{01} \\ 1 \\ 0 \end{pmatrix}$ is a corresponding eigenvector (provided $A_{00} - \alpha_{11} I$ is nonsingular).

TRUE ✔

- 일반화 : singular 하려면 diagonal 위치에 0이 있어야 한다. 왜 그렇지??

Consider the upper triangular matrix $U = \begin{pmatrix} \upsilon_{0,0} & \upsilon_{0,1} & \cdots & \upsilon_{0,n-1} \\ 0 & \upsilon_{1,1} & \cdots & \upsilon_{1,n-1} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \upsilon_{n-1,n-1} \end{pmatrix}$.

The eigenvalues of this matrix are $\upsilon_{0,0}, \upsilon_{1,1}, \ldots, \upsilon_{n-1,n-1}$.

Always ✔

Submit

An upper triangular matrix is invertible if and only if it has no zeros on the main diagonal. Here are some ways to see this:

1. The determinant of such a matrix is the product of the diagonal entries, and is non-zero if and only if the condition above holds.
2. The matrix has full rank whenever there are no zeros on the diagonal.
3. The inverse of the matrix can be explicitly computed via row operations. Use the bottom row to clean out the last column, the second to bottom row to clean out the second to last column, and so on.

12.1.3. 일반적인 행렬 예시

(1) 일단 왼쪽으로 넘기고

Find all (λ, x) (eigenpairs):

$$\begin{pmatrix} 1 & -1 \\ 2 & 4 \end{pmatrix} x = \lambda x$$

$$\left[\begin{pmatrix} 1 & -1 \\ 2 & 4 \end{pmatrix} - \lambda I\right] x = 0$$

$$\begin{pmatrix} 1-\lambda & -1 \\ 2 & 4-\lambda \end{pmatrix} x = 0$$

(2) determinant로 lambda구하기

(3) λ = 2, λ = 3를 해본다.

- $\lambda = 2$:

$$\begin{pmatrix} 1-2 & -1 \\ 2 & 4-2 \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

$$\begin{pmatrix} -1 & -1 \\ 2 & 2 \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \quad \Rightarrow \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} = \begin{pmatrix} -1 \\ 1 \end{pmatrix}.$$

- $\lambda = 3$:

$$\begin{pmatrix} 1-3 & -1 \\ 2 & 4-3 \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

$$\begin{pmatrix} -2 & -1 \\ 2 & 1 \end{pmatrix} \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \quad \Rightarrow \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} = \begin{pmatrix} -1 \\ 2 \end{pmatrix}.$$

(4) 복소수가 람다로 나오는 경우도 있다.

→ complex plane으로 visualization해야 한다.

- 3 x 3 matrix를 할 때에는 이것도 determinant를 만들면 된다. 똑같다.

- n x n matrix도 구하면 된다.

Determinants and Singular Matrices

- $A \in \mathbb{R}^{n \times n}$ is nonsingular if and only if $\det(A) \neq 0$.
- How is $\det(A)$ defined for $n > 3$?
- Characteristic polynomial:

$$p_n(\lambda) = \det(A - \lambda I)$$

- If $A \in \mathbb{R}^{n \times n}$, then $p_n(\lambda)$ has real coefficients:

$$p_n(\lambda) = \gamma_0 + \gamma_1 \lambda + \cdots + \gamma_{n-1}\lambda^{n-1} + \lambda^n,$$

where $\gamma_0, \ldots, \gamma_{n-1} \in \mathbb{R}$.

12.2. Diagonalization

- 핵심 : Eigenvalue를 찾는 것과 diagonalization과 깊은 관계가 있다는 것을 알 수 있다.
- 그러나, 모든 행렬이 diagonalize될 수 있는 것은 아니다.

12.2.1. Diagonalization의 방법과 효용성

Given $A \in \mathbb{R}^{n \times n}$, compute nonsingular X such that

$$X^{-1}AX = \Lambda,$$

where Λ is a diagonal matrix.

- 실제 diagonalization하는 방법 :

Find all (eigenpairs), (λ, x): $\begin{pmatrix} 1 & -1 \\ 2 & 4 \end{pmatrix} x = \lambda x$

$$\begin{pmatrix} 1 & -1 \\ 2 & 4 \end{pmatrix} \begin{pmatrix} -1 \\ 1 \end{pmatrix} = 2 \begin{pmatrix} -1 \\ 1 \end{pmatrix}$$

$$\begin{pmatrix} 1 & -1 \\ 2 & 4 \end{pmatrix} \begin{pmatrix} -1 \\ 2 \end{pmatrix} = 3 \begin{pmatrix} -1 \\ 2 \end{pmatrix}$$

$$\left(\begin{array}{cc|c} 1 & -1 & -1 \\ 2 & 4 & 1 \end{array}\right) \left(\begin{array}{c|c} -1 & -1 \\ 1 & 2 \end{array}\right) = \left(\begin{array}{c|c} 2\begin{pmatrix} -1 \\ 1 \end{pmatrix} & 3\begin{pmatrix} -1 \\ 2 \end{pmatrix} \end{array}\right)$$

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

$$\begin{pmatrix} -1 & -1 \\ 1 & 2 \end{pmatrix}^{-1} \begin{pmatrix} 1 & -1 \\ 2 & 4 \end{pmatrix} \begin{pmatrix} -1 & -1 \\ 1 & 2 \end{pmatrix} = \begin{pmatrix} 2 & 0 \\ 0 & 3 \end{pmatrix}$$

→ 와 람다로 이루어진 대각행렬이 되는구나!!!

- 효용성

Diagonal matrix로 표현하게 되면 장점은? : diagonal만 곱할 수 있기 때문에 좋다.

$$P^k = (X\Lambda X^{-1})^k = \underbrace{(X\Lambda X^{-1}) \times (X\Lambda X^{-1}) \times \cdots \times (X\Lambda X^{-1})}_{30 \text{ times}}$$

$$= X\Lambda \underbrace{X^{-1}X}_{I} \Lambda \underbrace{X^{-1}X}_{I} \cdots \Lambda X^{-1}$$

$$= X \underbrace{\Lambda \times \Lambda \times \cdots \times \Lambda}_{30 \text{ times}} X^{-1} = X\Lambda^k X^{-1}$$

$$\Lambda^k = \begin{pmatrix} \lambda_0 & 0 & \cdots & 0 \\ 0 & \lambda_1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \lambda_{n-1} \end{pmatrix}^k = \begin{pmatrix} \lambda_0^k & 0 & \cdots & 0 \\ 0 & \lambda_1^k & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \lambda_{n-1}^k \end{pmatrix}$$

12.2.1. Diagonalization될 수 있는 조건 : A가 n개의 linearly independent eigenvector가 있어야 한다.

- Let $A \in \mathbb{R}^{n\times n}$. Then A is diagonalizable if and only if there exists a nonsingular matrix X and diagonal matrix Λ such that $X^{-1}AX = \Lambda$.
- $X^{-1}AX = \Lambda \quad \Leftrightarrow \quad AX = X\Lambda \quad \Leftrightarrow \quad A = \Lambda X^{-1}$.
- A can be diagonalized if and only if A has n linearly independent eigenvectors.

- 증명 : 정방향의 증명 → $X^{-1}AX = \Lambda$ 이면 n개의 linearly independent eigenvalue가 있다.

Let $A \in \mathbb{R}^{n\times n}$. Then A can be diagonalized if and only if A has n linearly independent eigenvectors.

Proof:

$(\Rightarrow)$: $X^{-1}AX = \Lambda \quad \Rightarrow \quad AX = X\Lambda$.

$$X = \begin{pmatrix} x_0 \,|\, x_1 \,|\, \cdots \,|\, x_{n-1} \end{pmatrix} \text{ and } \Lambda = \begin{pmatrix} \lambda_0 & 0 & \cdots & 0 \\ 0 & \lambda_1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \lambda_{n-1} \end{pmatrix}.$$

$$\underbrace{AX}_{A\begin{pmatrix} x_0 \,|\, x_1 \,|\, \cdots \,|\, x_{n-1} \end{pmatrix}} = \underbrace{X\Lambda}_{\begin{pmatrix} x_0 \,|\, x_1 \,|\, \cdots \,|\, x_{n-1} \end{pmatrix}\begin{pmatrix} \lambda_0 & 0 & \cdots & 0 \\ 0 & \lambda_1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \lambda_{n-1} \end{pmatrix}}$$

$$\begin{pmatrix} Ax_0 \,|\, Ax_1 \,|\, \cdots \,|\, Ax_{n-1} \end{pmatrix} \qquad \begin{pmatrix} \lambda_0 x_0 \,|\, \lambda_1 x_1 \,|\, \cdots \,|\, \lambda_{n-1} x_{n-1} \end{pmatrix}$$

- 역방향의 증명 : $Ax_i = \lambda_i x_i$ 라면 $X^{-1}AX = \Lambda$이다.

Proof:

$(\Leftarrow)$: $Ax_i = \lambda_i x_i$

$$\underbrace{\begin{pmatrix} Ax_0 \,|\, Ax_1 \,|\, \cdots \,|\, Ax_{n-1} \end{pmatrix}}_{AX} = \underbrace{\begin{pmatrix} \lambda_0 x_0 \,|\, \lambda_1 x_1 \,|\, \cdots \,|\, \lambda_{n-1} x_{n-1} \end{pmatrix}}_{X\Lambda}$$

$$X^{-1}AX = \Lambda$$

- X가 역행렬이 없는 경우에는 대각행렬을 만들 수 없다. → 그런 경우를 Deficient matrix라고 한다.

The matrix $\begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}$ can be diagonalized.

TRUE ▾ ✖ **Answer:** FALSE

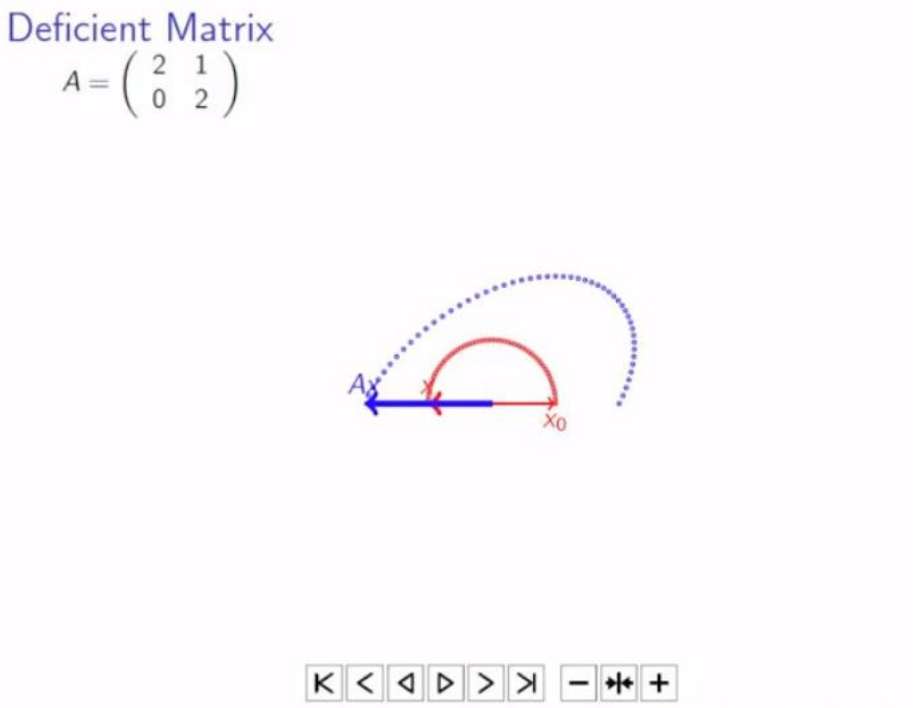
Since this matrix is upper triangular, we know that only the scalar $\lambda_0 = \lambda_1 = 0$ is an eigenvector. The problem is that the dimension of the null space of this matrix

$$\dim(\mathcal{N}(A - \lambda I)) = \dim(\mathcal{N}(\begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}) = 1.$$

Thus, we cannot find two linearly independent eigenvectors to choose as the columns of matrix X.

→ Ax=0에서 x 벡터가 선형독립이어야지 x벡터로 이루어진 X행렬이 역행렬이 존재하게 된다.

- 선생님 설명 : null space의 dimension은 number of free variable에 따른다. Null space에서 linearly independent vector는 1개 밖에 없다. 그러면 x벡터가 linearly independent 한 것이 1개 밖에 없기 때문에 X가 역행렬을 가질 수 없다.



→ 직관적인 설명 : 람다는 2이고, X를 구하면 (0 1 / 0 0)의 행렬이다. 그러나 이 행렬을 linearly independent한 vector가 1개 뿐이다. 왜냐하면 $\lambda_1 x_1 + \lambda_2 x_2 = 0$에서 λ가 모두 0이 되지 않아도 영벡터 앞의 λ1은 0이 아니어도 되기 때문에 linearly indepentant하지 않기 때문이다.

- Jordan block / Jordan factorization / Jordan canonical form

12.3. Determinant 심화

- 다차항의 determinant를 polynomial이라고 한다.

What we know about characteristic polynomials

- $\det(A-\lambda I) = p_n(\lambda) = \gamma_0 + \gamma_1\lambda + \cdots + \gamma_{n-1}\lambda^{n-1} + \lambda^n$
- n roots/eigenvalues, counting multiplicity.
  - k distinct roots/eigenvalues, with $k \le n$.
  - $\det(A-\lambda I) = p_n(\lambda) = (\lambda-\lambda_0)^{n_0}(\lambda-\lambda_1)^{n_1}\cdots(\lambda-\lambda_{k-1})^{n_{k-1}}$
  - $n_0 + n_1 + \cdots + n_{k-1} = n$.
  - n_j: (algebraic) multiplicity of root/eigenvalue λ_j.
- Even if $A \in \mathbb{R}^{n\times n}$ $(\gamma_0, \ldots, \gamma_{n-1} \in \mathbb{R})$
  - Some or all of the root /eigenvalues may be complex valued...
  - Complex roots/eigenvalues come in "conjugate pairs": If $\lambda = \lambda_R + i\lambda_C$ is a root, so is $\bar{\lambda} = \lambda_R - i\lambda_C$.
- Galois theory:
  For $n \ge 5$, roots of arbitrary $p_n(\lambda)$ cannot be found in a finite number of computations...

- 모든 polynomial determinant를 행렬로 표현할 수 있다.

$$p_n(\lambda) = \gamma_0 + \gamma_1\lambda + \cdots + \gamma_{n-1}\lambda^{n-1} + \lambda^n$$

$$A = \begin{pmatrix} -\gamma_{n-1} & -\gamma_{n-2} & \cdots & -\gamma_1 & -\gamma_0 \\ 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & 0 \end{pmatrix}$$

$\det(A-\lambda I)$

→ companion matrix : galois theory에 따르면 eigenvalue를 계산할 수 없다. 그래도 다른 방법이 있지는 않을까?

For $A \in \mathbb{R}^{n\times n}$

- The roots of $p_n(\lambda) = \det(A-\lambda I)$ are the eigenvalues of A.
- There are n eigenvalues, multiplicity counted.
- The *spectrum* of A, $\Lambda(A)$ is the set of all eigenvalues.
- A has $k = |\Lambda(A)|$ distinct eigenvalues.
- The algebraic multiplicity of eigenvalue λ_j equals the multiplity of the root λ_j of $p_n(\lambda)$.
- Eigenvalues can be complex. If so, they come in complex pairs.

질문!! : galois theory가 무엇인가??

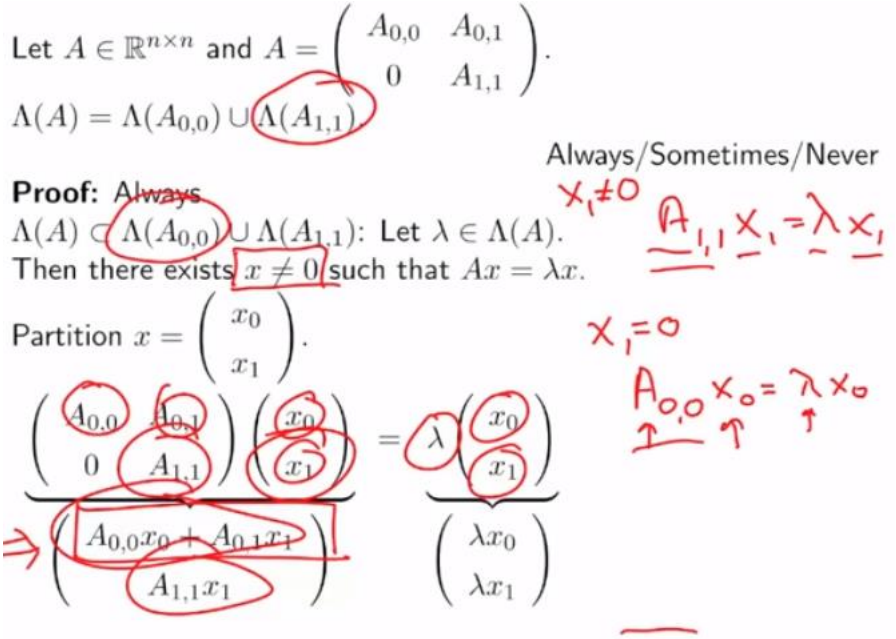
12.4. Eigenvalue, vector의 성질

- 성질1 : 행렬을 Upper triangular matrix랑 비슷하게 만들면 A를 구성하는 A0,0의 eigenvalue와 A11의 eigenvalue는 전체 행렬 A의 eigenvalue이기도 하다.

Let $A \in \mathbb{R}^{n \times n}$ and $A = \begin{pmatrix} A_{0,0} & A_{0,1} \\ 0 & A_{1,1} \end{pmatrix}$, where $A_{0,0}$ and $A_{1,1}$ are square matrices.

$$\Lambda(A) = \Lambda(A_{0,0}) \cup \Lambda(A_{1,1}).$$

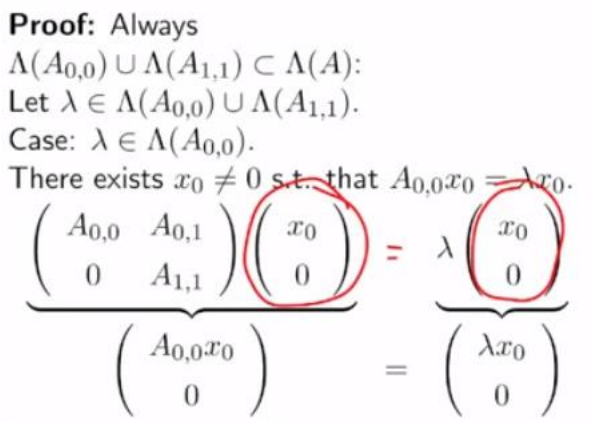
- 정방향 증명 :



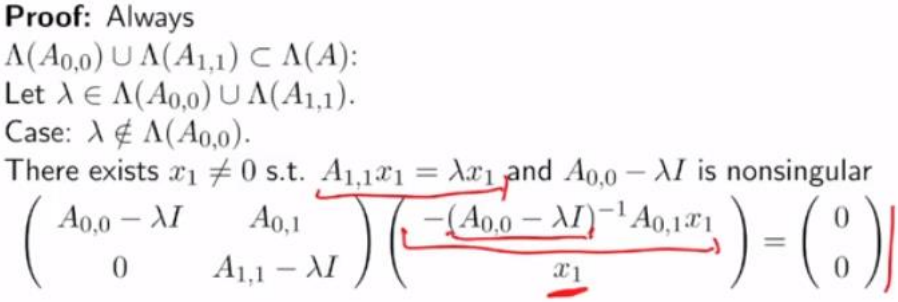
→ 쪼개서 x1 = 0, x1 !=0일 때로 나눠서 보면 된다.

- 역방향 증명 : 일단 λ ∈ Λ (A0,0) U Λ (A1,1)

1) 우선 만약에 람다가 A0,0의 eigenvalue라고 해보자.



2) 람다가 A0,0의 eigenvalue가 아니라고 해보자. A1,1 x1 = λ x1



→ 이를 통해서 Ax=0이 되는 x가 존재하는 것을 밝혀냈다. 즉 그 람다가 A의 eigenvalue가 되는구나!

- 일반화 : Block upper triangular matrix

Let $A \in \mathbb{R}^{n \times n}$ and

$$A = \begin{pmatrix} A_{0,0} & A_{0,1} & \cdots & A_{0,N-1} \\ 0 & A_{1,1} & \cdots & A_{1,N-1} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & A_{N-1,N-1} \end{pmatrix}.$$

Then $\Lambda(A) = \Lambda(A_{0,0}) \cup \Lambda(A_{1,1}) \cup \cdots \cup \Lambda(A_{N-1,N-1})$.

- 성질2 : A의 eigenvector는 내적이 0이다.

Let $A \in \mathbb{R}^{n \times n}$ be symmetric, $\lambda_i \neq \lambda_j$, $Ax_i = \lambda_i x_i$ and $Ax_j = \lambda_j x_j$.

$x_i^T x_j = 0$

(증명)

Let A be symmetric, $\lambda_i \neq \lambda_j$, $Ax_i = \lambda_i x_i$ and $Ax_j = \lambda_j x_j$.

$x_i^T x_j = 0$

Always/Sometimes/Never

$x_i^T A x_j = x_i^T \lambda_j x_j = \lambda_j x_i^T x_j$

$x_i^T A x_j = (A^T x_i)^T x_j = (Ax_i)^T x_j = (\lambda_i x_i^T) x_j = \lambda_i x_i^T x_j.$

$\lambda_j x_i^T x_j = \lambda_i x_i^T x_j$

$x_i^T x_j = 0$

- 성질3 :

If $\lambda \in \Lambda(A)$ then $\lambda \in \Lambda(A^T)$. True

(증명) 어려움

$\lambda \in \Lambda(A)$ $\Rightarrow$ $(A - \lambda I)$ is singular

$\Rightarrow$ < equivalent conditions >

$\dim(\mathcal{N}(A - \lambda I)) = k > 0$

$\Rightarrow$ < A is square, fundamental space picture >

$\dim(\mathcal{N}((A - \lambda I)^T)) = k > 0$

$\Rightarrow$ < $(A - \lambda I)^T = A^T - \lambda I$ >

$\dim(\mathcal{N}(A^T - \lambda I)) = k > 0$

$\Rightarrow$ < equivalent conditions >

$(A^T - \lambda I)$ is singular

$\Rightarrow$ < property of eigenvalue >

$\lambda \in \Lambda(A^T)$

→ transpose를 해도 linearly independent한 vector의 개수는 똑같으니까 dimension은 같다. 그리고 singular하려면 dimension of row space(=rank)가 n보다 작아야 한다. (k < n)

→ column space랑 row space의 dimension이 같은 이유가 무엇일까?

Proof. Suppose that $\{v_1, v_2, \ldots, v_k\}$ is a basis for the column space of A. Then each column of A can be expressed as a linear combination of these vectors; suppose that the i-th column c_i is given by

$$c_i = \gamma_{1i}v_1 + \gamma_{2i}v_2 + \cdots + \gamma_{ki}v_k$$

Now form two matrices as follows: B is an $m \times k$ matrix whose columns are the basis vectors v_i, while $C = (\gamma_{ij})$ is a $k \times n$ matrix whose i-th column contains the coefficients $\gamma_{1i}, \gamma_{2i}, \ldots, \gamma_{ki}$. It then follows[7] that $A = BC$.

However, we can also view the product $A = BC$ as expressing the *rows* of A as a linear combination of the rows of C with the i-th row of B giving the coefficients for the linear combination that determines the i-th row of A. Therefore, the rows of C are a spanning set for the row space of A, and so the dimension of the row space of A is at most k. We conclude that:

$$\dim(\text{rowsp}(A)) \leq \dim(\text{colsp}(A))$$

Applying the same argument to A^t, we conclude that:

$$\dim(\text{colsp}(A)) \leq \dim(\text{rowsp}(A))$$

and hence these values are equal

→ 질문! 이해가 안 된다.

- 날씨의 예에서 왜 계속 날씨 행렬을 곱해도 벡터가 똑같은지를 알아보자.

Consider

$$x^{(k+1)} = Px^{(k)}$$

Why does $x^{(k)}$ eventually have the property that $Px^{(k)} \approx x^{(k)}$?

$$\Lambda\left(\begin{pmatrix} 0.4 & 0.3 & 0.1 \\ 0.4 & 0.3 & 0.6 \\ 0.2 & 0.4 & 0.3 \end{pmatrix}\right) = \{1, \lambda_1, \lambda_2\} \quad \text{with} \quad 1 > \lambda_1 \geq \lambda_2$$

- Let $(1, v_0)$, (λ_1, v_1), and (λ_2, v_2) be eigenpairs.
- Assume that v_0, v_1, v_2 are linearly independent. (They are.)
- $x^{(0)} = \gamma_0 v_0 + \gamma_1 v_1 + \gamma_2 v_2$
- $x^{(1)} = Px^{(0)} = P(\gamma_0 v_0 + \gamma_1 v_1 + \gamma_2 v_2)$
  $= \gamma_0 \underbrace{Pv_0}_{v_0} + \gamma_1 \underbrace{Pv_1}_{\lambda_1 v_1} + \gamma_2 \underbrace{Pv_2}_{\lambda_2 v_2}$

- $x^{(0)} = \gamma_0 v_0 + \gamma_1 v_1 + \gamma_2 v_2$
- $x^{(1)} = Px^{(0)} = P(\gamma_0 v_0 + \gamma_1 v_1 + \gamma_2 v_2)$
  $= \gamma_0 \underbrace{Pv_0}_{v_0} + \gamma_1 \underbrace{Pv_1}_{\lambda_1 v_1} + \gamma_2 \underbrace{Pv_2}_{\lambda_2 v_2}$
- $x^{(2)} = Px^{(1)} = P(\gamma_0 v_0 + \gamma_1 \lambda_1 v_1 + \gamma_2 \lambda_2 v_2)$
  $= \gamma_0 \underbrace{Pv_0}_{v_0} + \gamma_1 \underbrace{P\lambda_1 v_1}_{\lambda_1^2 v_1} + \gamma_2 \underbrace{P\lambda_2 v_2}_{\lambda_2^2 v_2}$

$$\lim_{k\to\infty} x^{(k)} = \lim_{k\to\infty} \left(\gamma_0 v_0 + \gamma_1 \lambda_1^k v_1 + \gamma_2 \lambda_2^k v_2\right)$$

$$= \gamma_0 v_0 + \gamma_1 \left(\lim_{k\to\infty} \lambda_1^k\right) v_1 + \gamma_2 \left(\lim_{k\to\infty} \lambda_2^k\right) v_2$$

$$= \gamma_0 v_0 + \gamma_1 (0) v_1 + \gamma_2 (0) v_2$$

cf) The power method : 왜 하는지 모르겠다!

- $x^{(0)} = \gamma_0 v_0 + \gamma_1 v_1 + \cdots + \gamma_{n-1} v_{n-1}$
- $x^{(1)} = Ax^{(0)} = A(\gamma_0 v_0 + \gamma_1 v_1 + \cdots + \gamma_{n-1} v_{n-1})$
  $= \gamma_0 \underbrace{Av_0}_{\lambda_0 v_0} + \gamma_1 \underbrace{Av_1}_{\lambda_1 v_1} + \cdots + \gamma_{n-1} \underbrace{Av_{n-1}}_{\lambda_{n-1} v_{n-1}}$
- $x^{(2)} = Ax^{(1)} = A(\gamma_0 \lambda_0 v_0 + \gamma_1 \lambda_1 v_1 + \cdots + \gamma_{n-1} \lambda_{n-1} v_{n-1})$
  $= \gamma_0 \underbrace{A\lambda_0 v_0}_{\lambda_0^2 v_0} + \gamma_1 \underbrace{A\lambda_1 v_1}_{\lambda_1^2 v_1} + \cdots + \gamma_{n-1} \underbrace{A\lambda_{n-1} v_{n-1}}_{\lambda_{n-1}^2 v_{n-1}}$
  $\vdots$
- $x^{(k+1)} = Ax^{(k)} = A(\gamma_0 \lambda_0^k v_0 + \gamma_1 \lambda_1^k v_1 + \cdots + \gamma_{n-1} \lambda_{n-1}^k v_{n-1})$
  $= \gamma_0 \underbrace{A\lambda_0^k v_0}_{\lambda_0^{k+1} v_0} + \gamma_1 \underbrace{A\lambda_1^k v_1}_{\lambda_1^{k+1} v_1} + \cdots + \gamma_{n-1} \underbrace{A\lambda_{n-1}^k v_{n-1}}_{\lambda_{n-1}^{k+1} v_{n-1}}$
  $= \gamma_0 \lambda_0^{k+1} v_0 + \gamma_1 \lambda_1^{k+1} v_1 + \cdots \gamma_{n-1} \lambda_{n-1}^{k+1} v_{n-1}$

$$x^{(0)} = \gamma_0 v_0 + \gamma_1 v_1 + \cdots + \gamma_{n-1} v_{n-1}$$

$$x^{(1)} = \frac{1}{\lambda_0} A x^{(0)} = \frac{1}{\lambda_0} A(\gamma_0 v_0 + \gamma_1 v_1 + \cdots + \gamma_{n-1} v_{n-1})$$

$$= \gamma_0 \underbrace{\frac{1}{\lambda_0} A v_0}_{v_0} + \gamma_1 \underbrace{\frac{1}{\lambda_0} A v_1}_{\frac{\lambda_1}{\lambda_0} v_1} + \cdots + \gamma_{n-1} \underbrace{\frac{1}{\lambda_0} A v_{n-1}}_{\frac{\lambda_{n-1}}{\lambda_0} v_{n-1}}$$

$$x^{(2)} = \frac{1}{\lambda_0} A x^{(1)} = \frac{1}{\lambda_0} A(\gamma_0 v_0 + \gamma_1 \frac{\lambda_1}{\lambda_0} v_1 + \cdots + \gamma_{n-1} \frac{\lambda_{n-1}}{\lambda_0} v_{n-1})$$

$$= \gamma_0 \underbrace{\frac{1}{\lambda_0} A v_0}_{v_0} + \gamma_1 \underbrace{\frac{1}{\lambda_0} A \frac{\lambda_1}{\lambda_0} v_1}_{\left(\frac{\lambda_1}{\lambda_0}\right)^2 v_1} + \cdots + \gamma_{n-1} \underbrace{\frac{1}{\lambda_0} A \frac{\lambda_{n-1}}{\lambda_0} v_{n-1}}_{\left(\frac{\lambda_{n-1}}{\lambda_0}\right)^2 v_{n-1}}$$