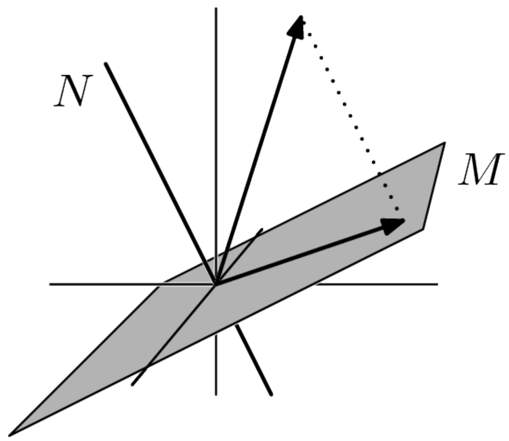

DeepXE

Course: Linear Algebra

January 13, 2023

Author	*Student ID*
Jonghyeok Kim	20201967

Linear Algebra (1/8)

Jonghyeok Kim

January 13, 2023

Contents

1 Linear Equations and Matrices **4**

1.1 Introduction to Linear Algebra 4

1.1.1 System of linear equations 4

1.1.2 Types of solutions 4

1.2 Gaussian Elimination 5

1.2.1 Introduction to matrices 5

1.2.2 Elementary row operations 5

1.2.3 Gaussian elimination 6

1.2.4 Extending row operations 7

1.3 The Transpose and Inverse of a Matrix 7

1.3.1 Transpose of a matrix 8

1.3.2 Properties of matrix tranpose 9

1.4 The Inverse Matrix Method 10

1.4.1 Elementary matrices 10

1.4.2 Equivalent statements 12

2 Euclidean Space **16**

2.1 Properties of Vectors 16

2.1.1 Vector addition and scalar multiplication properties 16

2.1.2 Dot(inner) product revisited 18

2.1.3 Angle between two vectors 20

2.1.4 Inequalities 20

2.1.5 Properties of linear dependence 20
2.1.6 Basis 22
2.1.7 Properties of bases 22

3 Conclusion **24**
3.1 Conclusion 24

4 Acknowledgement **24**

A Types of Matrices **25**
A.1 Tutorial Overview 25
A.1.1 Square Matrix 25
A.1.2 Symmetric Matrix 26
A.1.3 Triangular Matrix 27

B References **30**

1 Linear Equations and Matrices

1.1 Introduction to Linear Algebra

Linear Algebra는 Linear Equations들에 대한 Study이다.

여기서 Linear Equation이란, 모든 변수들 (ex. $x, y, z, \cdots$)이 0 또는 1의 지수를 가지는 것이다.

만약에 어떤 방정식이 Trigonometric, Exponential, Logarithmic 또는 Hyperbolic 함수들의 Argument를 포함하고 있다면 이 경우 Linear하다고 하지 않는다. (ex. $sin(x) + y + z = 1$)

1.1.1 System of linear equations

참고로 Linear Equation의 집합을 System of Linear Equations 또는 Linear System이라 한다. 단, Linear Equations들의 수가 유한해야 하고, 유한 개의 미지수를 가져야 한다.

$$\begin{aligned} a_{11}x_1 \,+\, a_{12}x_2 \,+\cdots+\, a_{1n}x_n \,&=\, b_1 \\ a_{21}x_1 \,+\, a_{22}x_2 \,+\cdots+\, a_{2n}x_n \,&=\, b_2 \\ \vdots \qquad\quad \vdots \qquad \ddots \qquad \vdots \quad &\quad\ \vdots \\ a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n &= b_m \end{aligned} \tag{1}$$

식 (1)에서 a_{ij}, b_i는 실수(real number)이고, $x_1, x_2, x_3, \cdots, x_n$은 실수에 대한 Placeholder이다.

1.1.2 Types of solutions

선형대수학(linear algebra)는 Section 1.1에서 상술하였듯, Linear Equations들에 대한 Study이므로, 이러한 Linear Equations들에 대한 해(solution)가 존재하는지/존재하지 않는지, 만약 존재한다면 얼마나 많은 해가 존재하는지에 대한 근본적인 의문들을 가진다.

만약 Linear System이 해를 가지지 않는다면, **inconsistent**하다고 말한다.

반면에, Linear System이 최소한 하나 이상의 해를 가진다면, **consistent**하다고 말한다.

해(solution)는 다음과 같이 크게 세 가지 종류로 구분할 수 있다.

- No solution
- Unique solution
- Infinite number of solution

1.2 Gaussian Elimination

1.2.1 Introduction to matrices

다음과 같은 Linear system이 있다고 해보자.

$$\begin{cases} x + 2y = 3 \\ 2x + y = 2.5 \end{cases} \tag{2}$$

위의 식 (2)의 좌변의 첫 번째 열(column)은 미지수 x의 계수들만이 존재하고, 두 번째 열은 미지수 y의 계수들만이 존재한다.

그러므로, 이러한 미지수의 계수들을 다음과 같은 Matrix로 표현할 수 있다.

$$\begin{pmatrix} 2 & 2 \\ 2 & 1 \end{pmatrix} \tag{3}$$

또한, 식 (2)의 등호 기준 우변에는 좌변과 달리 하나의 열(column)만이 존재하기 때문에 이를 다음과 같이 하나의 벡터(vector)로 나타낼 수 있다.

$$\begin{pmatrix} 3 \\ 2.5 \end{pmatrix} \tag{4}$$

참고로 Vector는 Matrix의 한 종류이다. 즉, Vector는 Matrix이다.

이렇듯 Matrix는 데이터를 저장하기 위한 효율적인 방법이다.

1.2.2 Elementary row operations

앞서 살펴본 식 (1)은 Section 1.2.1에서 다룬 Matrix를 이용하여 다음과 같이 표현할 수 있다.

$$\left(\begin{array}{ccc|c} a_{11} & \cdots & a_{1n} & b_1 \\ \vdots & \ddots & \vdots & \vdots \\ a_{m1} & \cdots & a_{mn} & b_m \end{array}\right) \tag{5}$$

위의 Matrix (5)는 Augmented matrix인데, 수직선 기준 좌변은 미지수 $x_1, x_2, x_3, \cdots, x_n$의 계수(coefficients)들이 오고, 우변은 Linear equation의 등호 기준 우변에 있는 상수(constant) 값이 온다. 즉, 앞서 Section 1.2.1에

서 살펴본 Matrix (3)와 (4)를 하나의 Matrix로 표현한 것이 바로 이 Augmented Matrix이다.

여기서, Augmented란 'To increase'를 의미하는데, 다시 말해서 '증가되어진', '추가되어진'이라는 의미로 직역할 수 있다.

즉, 기존 Matrix가 다음과 같이 표현되는데 반해,

$$\begin{pmatrix} 1 & 1 & 1 \\ 2 & 2 & 2 \\ 3 & 3 & 3 \end{pmatrix} \tag{6}$$

Augmented matrix는 위의 기존 Matrix (6)에 '수직선(vertical line)'이 추가되어진 것이다.

$$\left(\begin{array}{ccc|c} 1 & 1 & 1 & 1 \\ 2 & 2 & 2 & 2 \\ 3 & 3 & 3 & 3 \end{array}\right) \tag{7}$$

이러한 Augmented matrix는 이전에 살펴보았던 Matrix (5)에서와 같이 어떤 Linear system of equations을 짧고 간략하게 표현하는 데 사용된다.

1.2.3 Gaussian elimination

다음과 같은 Linear system이 있다고 해보자.

$$\begin{cases} x - 3y + 5z = -9 \\ 2x - y - 3z = 19 \\ 3x + y + 4z = -13 \end{cases} \tag{8}$$

식 (8)을 다음과 같이 변환하는 과정을 Gaussian Elimination with Back substitution이라 한다.

$$\begin{cases} x - 3y + 5z = -9 \\ 5y - 13z = 37 \\ 15z = -60 \end{cases} \tag{9}$$

어떻게 보면, Upper triangular matrix 즉 Main diagonal 기준, Bottom part를 0으로 만드는 과정으로 생각할 수도 있겠다.

1.2.4 Extending row operations

최종적으로, 다음과 같은 Augmented Matrix로 변환하는 것이 목표이다.

$$\left(\begin{array}{ccc|c} 1 & 0 & 0 & * \\ 0 & 1 & 0 & * \\ 0 & 0 & 1 & * \end{array}\right) \tag{10}$$

위와 같은 Augmented matrix를 Reduced row echelon 형태에 있다고 한다.

어떠한 행렬이 Reduced row echolon 형태에 있기 위해서는 다음의 조건들을 모두 만족시켜야 한다.

1. 0만을 포함하고 있는 행들이 Matrix의 Bottom part에 위치해 있어야 한다.
2. 어떤 행이 0이 아닌 Entries들을 가지고 있는 데, 첫 번째로 오는 0이 아닌 Entry는 1이어야 한다. (이때 1을 Leading 1이라 한다.)
3. 두 개의 연이은 0이 아니고, Leading 1의 Entries를 가지고 있는 행들이 Top left $\sim$ Bottom right상에 위치해 있어야 한다.
4. Leading 1을 포함하고 있는 어떤 열에서 0이 아닌 Entry가 오직 Leading 1 뿐이어야 한다.

위의 조건들 중 마지막 4번의 조건이 만족되지 않고 나머지 조건들만 만족된다면, 이 경우 해당 Augmented matrix가 Row echelon 형태에 있다고 한다.

참고로, 조건에서 상술하였듯 첫 번째로 오는 1의 Entry를 두고 Leading 1이라 하는데, 일부 선형대수학 문헌에서는 0이 아닌 어떠한 Leading 숫자들에 대해 Leading coefficient라는 용어를 사용하고 있다.

만약 어떠한 Augmented Matrix를 Row echolon 형태로 변환하는 과정을 Gaussian Elimination이라 하고,

Reduced row echolon 형태로 변환하는 과정은 Gauss-Jordan elimination이라 한다.

결론적으로, Augmented Matrix를 Reduced row echolon 형태로 변환하는 목적은 Back substitution을 피하기 위해서이다.

추후에 증명하겠지만, 만약 어떠한 Augmented matrix가 주어진다면, 그것의 Reduced row echelon 형태는 유일하나, Row echelon 형태는 유일하지 않다.

1.3 The Transpose and Inverse of a Matrix

Matrix는 실수(Real number)에서와 같이 나누기 연산이 불가능하지만, 이것과 가장 유사한 연산으로서 **Inverse** 연산이 존재한다.

이러한 Inverse Matrix를 통해 Linear Systems의 해를 찾는 것이 가능하다.

뒤에서도 살펴보겠지만, Inverse 연산 외에 또 다른 중요한 연산으로서 **Transpose** 연산이 존재한다.

1.3.1 Transpose of a matrix

Transpose 연산은 주어진 Matrix의 행(Rows)을 회전시킨 새로운 Matrix를 만들어주는 연산이다.

예를 들어, 다음과 같은 Matrix가 있다고 해보자.

$$\begin{pmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{pmatrix} \tag{11}$$

위 Matrix (11)의 Transpose는 다음과 같다.

$$\begin{pmatrix} 1 & 3 & 5 \\ 2 & 4 & 6 \end{pmatrix} \tag{12}$$

위 예제를 통해 확인할 수 있듯이, Column 1은 Row 1이 되고, Column 2는 Row 2가 된다.

특히, A의 Transpose는 A^T로 표기할 수 있다.

일반적으로, Matrix A의 entry인 a_{ij}는 A^T 내의 entry인 a_{ji}로 Transpose 되어진다.

앞서 Linear systems의 미지수(Unknown)의 벡터(Vector)인 $x = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix}$는 Column 벡터로서, 이것을 Transpose 하게 되면, Row 벡터가 되므로, 이를 Row 벡터의 Transpose로 다음과 같이 나타낼 수 있다.

$$\mathbf{x} = \begin{pmatrix} x_1 & \cdots & x_n \end{pmatrix}^T \tag{13}$$

이렇듯 Column 벡터를 Row 벡터로 표기하는 것은 공간을 절약할 수 있는데, 위 경우에는 Column 벡터 $\mathbf{x}$가 n개의 Lines을 차지하는 데 반해, $\mathbf{x} = \begin{pmatrix} x_1 & \cdots & x_n \end{pmatrix}^T$는 1개의 Line만을 사용할 수 있다.

중요한 것은 $m \times n$의 Matrix를 Transpose하게 되면, $n \times m$의 Matrix가 도출된다는 사실이다.

만약 $n \neq m$이면, Tranpose 연산으로 Matrix의 모양(shape)을 변화시킬 수 있다.

1.3.2 Properties of matrix tranpose

Theorem 1. *Properties of matrix transpose*
A와 B가 아래의 연산(*Operations*)들이 수행될 수 있도록 적절한 크기(*An appropriate size*)를 가진 *Matrices*라고 해보자. 여기서, k는 스칼라(*Scalar*)이다.

(*a*) $(A^T)^T = A$

(*b*) $(kA)^T = kA^T$

(*c*) $(A+B)^T = A^T + B^T$

(*d*) $(AB)^T = B^T A^T$

위 Theorem 1 중 (a)의 의미는 다음과 같다.

Figure 1: The meaning of $(A^T)^T$

즉, Transpose 처리된 Matrix를 다시 Transpose 연산을 적용하게 되면, 처음 시작했던 Matrix가 된다는 의미이다.

이를 Top left에서 Bottom right까지의 Diagonal에 대해 Matrix를 뒤집은 다음, 다시 Diagonal에 대해 뒤집으면, 처음의 Matrix가 다시 도출되는 것으로 생각해볼 수 있다.

이를 증명해보자.

Proof 1

a_{ij}를 Matrix A의 i번째 행과 j번째 열에 있는 Entry라 하자.

그러면 이는 다음과 같다.

$$A^T = (a_{ij})^T = (a_{ji}) \tag{14}$$

위의 식 (14)에 대해 또 다시 Transpose를 취하게 되면 다음과 같다.

$$(A^T)^T = (a_{ji})^T = (a_{ij}) = A \;\square \tag{15}$$

위 Proof 1에서 Entries들은 두 번 뒤바뀌었는데, 예를 들어, $a_{21} \to a_{12} \to a_{21}$, $a_{31} \to a_{13} \to a_{31}$ 이다.

그러므로, $(A^T)^T = A$임이 증명되었다.

이번에는 Theorem 1 중 (c)를 증명해보자.

Proof 2

$$
\begin{aligned}
(A+B)^T &= \left[\begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{pmatrix} + \begin{pmatrix} b_{11} & b_{12} & \cdots & b_{1n} \\ b_{21} & b_{22} & \cdots & b_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ b_{m1} & b_{m2} & \cdots & b_{mn} \end{pmatrix}\right]^T \\
&= \begin{pmatrix} a_{11}+b_{11} & a_{12}+b_{12} & \cdots & a_{1n}+b_{1n} \\ a_{21}+b_{21} & a_{22}+b_{22} & \cdots & a_{2n}+b_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ a_{m1}+b_{m1} & a_{m2}+b_{m2} & \cdots & a_{mn}+b_{mn} \end{pmatrix}^T \\
&= \begin{pmatrix} a_{11}+b_{11} & a_{21}+b_{21} & \cdots & a_{m1}+b_{m1} \\ a_{12}+b_{12} & a_{22}+b_{22} & \cdots & a_{m2}+b_{m2} \\ \vdots & \vdots & \vdots & \vdots \\ a_{1n}+b_{1n} & a_{2n}+b_{2n} & \cdots & a_{mn}+b_{mn} \end{pmatrix} \\
&= \begin{pmatrix} a_{11} & a_{21} & \cdots & a_{m1} \\ a_{12} & a_{22} & \cdots & a_{m2} \\ \vdots & \vdots & \vdots & \vdots \\ a_{1n} & a_{2n} & \cdots & a_{mn} \end{pmatrix} + \begin{pmatrix} b_{11} & b_{21} & \cdots & b_{m1} \\ b_{12} & b_{22} & \cdots & b_{m2} \\ \vdots & \vdots & \vdots & \vdots \\ b_{1n} & b_{2n} & \cdots & b_{mn} \end{pmatrix} = A^T + B^T \; \square
\end{aligned} \tag{16}
$$

1.4 The Inverse Matrix Method

1.4.1 Elementary matrices

Elementary matrix는 Identity Matrix I에 대해 Single 행 연산에 의해 구해지는 Matrix이다.

예를 들어, 다음은 Elementary matrix인데,

$$
\begin{pmatrix} 1 & 0 & 0 \\ 0 & \sqrt{2} & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{17}
$$

그 이유는 Identity matrix I의 중간 행에 $\sqrt{2}$를 곱했기 때문이다.

$$\begin{pmatrix} 1 & 0 & 3 \\ 0 & 4 & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{18}$$

위 Matrix (18)는 Elementary matrix가 아닌데, 그 이유는 2개의 Row operations이 수반되기 때문이다.

즉 다음의 2개의 Row operations이 수반되는 것이다.

1. Identity matrix의 아래 행에 3을 곱하고 제일 위의 행과 더한다.
2. Identity matrix의 중간 행을 4로 곱한다.

부가적으로, 다음은 Elementary matrix인데,

$$\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \tag{19}$$

그 이유는 행을 뒤바꾸는(Swapping) 연산 역시 Single row operation에 해당하기 때문이다.

만약 어떤 Elementary matrix E가 특정한 행 연산에 의해서 Identity matrix에 의해서 구해진다면, 그 Matrix 곱셈 EA는 Matrix A에 대해서도 동일한 연산을 수행한다.

예를 들어, 다음과 같은 아래 두 개의 행들을 서로 교환함으로써 Identity matrix에 의해서 구해진 3×3의 일반적인 Matrix를 고려해보자.

$$A = \begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix} \text{ and } E = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix} \tag{20}$$

위의 Matrix E는 Size 3의 Identity Matrix의 가장 아랫쪽 행과 중간의 행을 서로 맞바꾸는 단일의 행 연산 (Single row operations)이 진행된 후의 Elementary matrix이다.

그러므로, Matrix 곱셈 EA는 다음과 같다.

$$EA = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix} = \begin{pmatrix} a & b & c \\ g & h & i \\ d & e & f \end{pmatrix} \tag{21}$$

위의 Matrix 곱셈 (21)을 살펴보면, 곱셈의 결과에 적용된 연산이 위의 Matrix E에 적용된 연산과 동일한 것을 확인해볼 수 있다.

이러한 Idea를 응용하여 다음과 같은 Elementary matrix E가 있을 때,

$$E = \begin{pmatrix} 1 & 0 & 0 \\ 0 & -2 & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{22}$$

이러한 Matrix (22)는 Size 3의 Identity matrix의 중간 행에 -2를 곱함으로써 구해지기 때문에, 다음과 같은 Matrix 곱셈 EA의 결과가 도출되는 것은 지극히 자연스럽다.

$$EA = \begin{pmatrix} 1 & 0 & 0 \\ 0 & -2 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix} = \begin{pmatrix} a & b & c \\ -2d & -2e & -2f \\ g & h & i \end{pmatrix} \tag{23}$$

이를 일반화하여, 만약 E가 Identity matrix I_n에 대해 특정한 row 연산을 수행함으로써 구해진 $n \times n$의 Elementary matrix이고, A는 $n \times r$의 Matrix이라면, Matrix 곱셈 EA는 Matrix A에 동일한 row 연산을 수행하게 된다. 그러므로 Row 연산을 수행하기 위해 Matrix E를 사용할 수 있다.

1.4.2 Equivalent statements

Equivalence란 무엇을 의미하는 것일까?

두 개의 명제(Statements)들 P와 Q가 Equivalent하다는 것은 만약 P와 Q가 모두 사실(True)이거나 거짓(False)인 경우에 성립한다.

이것을 표현하는 또 다른 방식으로 P와 Q는 동치(if and only if)이다고 하며, $P \iff Q$로 표기된다.

즉, 이것은 P implies Q와 Q implies P를 의미하고, $P \implies Q$ 그리고 $Q \implies P$로 표기된다.

추가적으로, 다음과 같은 형태의 명제(Proposition)을 증명함에 있어,

$$P \implies Q \tag{24}$$

P가 참이라고 가정한 상황에서 Q를 추론(Deduce)해내면 된다.

일반적으로, 만약 $P \implies Q \implies R \implies S \implies P$라면 이러한 4개의 명제들은 Equivalent하다고 한다.

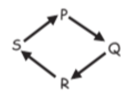

Figure 2: Equivalence

다음으로, Invertible matrices에 대한 중요한 정리(Theorem)를 증명할 것이다.

Theorem 2. *A가 $n \times n$의 Matrix라 하자. 그러면 다음의 4개의 명제들은 Equivalent하다.*

(a) *Matrix A는 Invertible(Non-singular)하다.*

(b) *Linear system $A\boldsymbol{x} = O$는 오직 Trivial solution 즉 $\boldsymbol{x} = O$를 가진다.*

(c) *Matrix A의 Reduced row echelon 형태는 Identity matrix I이다.*

(d) *A는 Elementary matrices들의 곱셈(Product)이다.*

앞서 Figure 2에서 살펴본 것과 같이, 위의 정리 2는 $(a) \implies (b) \implies (c) \implies (d) \implies (a)$로 증명할 것이다.

먼저 $(a) \implies (b)$부터 증명해보자.

Proof 3

우선 명제 (a)가 사실이라고 가정하고, 명제 (b)를 추론해낸다.

A가 Invertible matrix라고 가정함으로써, Linear system $A\mathbf{x} = O$가 오직 Trivial solution $\mathbf{x} = O$를 가지는 것을 보이면 된다.

Linear system $A\mathbf{x} = O$를 고려해보자.

A는 Invertible하기 때문에 $A^{-1}A = I$를 만족하는 유일한 A^{-1} Matrix가 존재한다.

$A\mathbf{x} = O$의 양변을 A^{-1}로 곱한 결과는 다음과 같다.

$$A^{-1}A\mathbf{x} = A^{-1}O \tag{25}$$

$$I\mathbf{x} = A^{-1}O = O \tag{26}$$

$$I\mathbf{x} = O \tag{27}$$

$$\therefore\ \mathbf{x} = O \qquad [\because\ I\mathbf{x} = \mathbf{x}] \qquad \square \tag{28}$$

$\mathbf{x} = O$라는 결론은 오직 Trivial solution $\mathbf{x} = O$를 가짐을 의미한다.

다음으로 $(b) \implies (c)$를 증명하자.

Proof 4

이것 역시 명제 (b)가 참이라는 가정 하에 명제 (c)를 추론해내면 증명이 완료된다.

$A\mathbf{x} = O$가 오직 Trivial solution $\mathbf{x} = O$를 가진다고 가정한 상황에서 Matrix A의 Reduced row echelon 형태가 Identity matrix I임을 증명하자.

Matrix A의 Reduced row echelon 형태는 0으로만 구성된 행(Rows; Equations)을 가질 수 없다.

만약 0으로만 구성된 행을 가지게 된다면, 미지수의 개수 n가 0이 아닌 행들(방정식들)의 개수 r보다 더 클 것이다.

이러한 경우 수식으로 표현해보면, $r < n$이고, Linear system $A\mathbf{x} = O$는 무한히 많은 해(Solutions)를 가지게 된다.

그러나, $A\mathbf{x} = O$에는 오직 하나의 해(Unique solution)인 $\mathbf{x} = O$만을 가지기 때문에 이러한 가정은 성립하지 않는다. 즉 $r \not< n$인 것이다. 다시 말해서, 0으로만 구성된 행을 가지지 않는다는 의미이다.

그런데, 다음 명제(Proposition)에 의하면,

Proposition 1. 어떠한 *Reduced row echelon* 형태에 있는 *Matrix*는 *Identity matrix* I이거나, *0*으로만 구성된 행을 가지고 있다.

이미 위에서 0으로만 구성된 행을 가지지 않음을 증명하였기 때문에, 결국 Matrix A의 Reduced row echelon 형태는 Identity matrix I이다. $\square$

$(c) \implies (d)$를 증명하자.

Proof 5

(c)가 참이라는 가정 하에 (d)를 추론해낸다.

즉, Matrix A의 Reduced row echelon 형태는 Identity matrix I인 것이 사실인 상황에서 Matrix A가 Elementary matrices들의 곱으로 표현됨을 보이면 된다.

Matrix A의 Reduced row echelon 형태가 Identity matrix I라는 것은 Matrix A가 Identity matrix I에 Row equivalent하다는 의미이다.

즉, 다음의 정의에 의하여,

Definition 1. B는 *Matrix* A에 *Row equivalent*하다는 것은 $B = E_n E_{n-1} \cdots E_2 E_1 A$와 동치임을 뜻한다.

다음을 만족하는 Elementary matrices $E_1, E_2, \cdots E_k$가 있다.

$$A = E_k E_{k-1} \cdots E_2 E_1 I \qquad [\because \ A \text{ is row equivalent to } I] \tag{29}$$

$$= E_k E_{k-1} \cdots E_2 E_1 \qquad [\because \ I \text{ is the identity matrix}] \qquad \square \tag{30}$$

그러므로, Matrix A가 이러한 Elementary matrices들의 곱(Product)으로 표현됨을 보였다.

$(d) \implies (a)$를 증명하자.

Proof 6 Matrix A가 Elementary matrices들의 곱으로 표현된다는 가정 하에, Matrix A가 Invertible함을 보이자.

기본적으로, Elementary matrices들은 Invertible하다 즉, Inverse를 가지고 있다. 그러므로 Matrix 곱셈 $E_k E_{k-1} \cdots E_2 E_1$ 역시 Invertible하다.

Matrix A의 양변에 Inverse를 취하면 다음과 같다.

$$A^{-1} = (E_k E_{k-1} \cdots E_2 E_1)^{-1} \tag{31}$$

$$= E_1^{-1} E_2^{-1} \cdots E_{k-1}^{-1} E_k^{-1} \qquad [\because \ (XYZ)^{-1} = Z^{-1} Y^{-1} X^{-1}] \tag{32}$$

Invertible(Non-singular)의 정의를 만족시킴을 확인해보자.

$$A^{-1} A = (E_1^{-1} E_2^{-1} \cdots E_{k-1}^{-1} E_k^{-1})(E_k E_{k-1} \cdots E_2 E_1) = I \tag{33}$$

$$A A^{-1} = (E_k E_{k-1} \cdots E_2 E_1)(E_1^{-1} E_2^{-1} \cdots E_{k-1}^{-1} E_k^{-1}) = I \qquad \square \tag{34}$$

그러므로 A는 Invertible(Non-singular)하다.

결론적으로, $(a) \implies (b) \implies (c) \implies (d) \implies (a)$를 증명했다. 즉 (a), (b), (c), (d)는 모두 Equivalent함을 증명했다.

2 Euclidean Space

2.1 Properties of Vectors

2.1.1 Vector addition and scalar multiplication properties

R^n은 Euclidean n-space이고, 그리스 수학자 Euclid 사후에 명명되었다.

Proposition 2. $\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w}$가 R^n에 있는 벡터이고, k, c가 실수(또는 실수 스칼라)라 하자.

1. $\boldsymbol{u} + \boldsymbol{v} = \boldsymbol{v} + \boldsymbol{u}$ (*commutative law*)
2. $(\boldsymbol{u} + \boldsymbol{v}) + \boldsymbol{w} = \boldsymbol{u} + (\boldsymbol{v} + \boldsymbol{w})$ (*associative law*)
3. $\boldsymbol{u} + \boldsymbol{O} = \boldsymbol{u}$를 만족시키는 *Zero vector* $\boldsymbol{O}$가 존재한다.
4. 모든 벡터 $\boldsymbol{u}$에 대해서 $\boldsymbol{u} + (-\boldsymbol{u}) = \boldsymbol{O}(additiveinverse)$를 만족하는 벡터 $-\boldsymbol{u}$가 존재한다.
5. $k(\boldsymbol{u} + \boldsymbol{v}) = k\boldsymbol{u} + k\boldsymbol{v}$ (*distributive law*)
6. $(k + c)\boldsymbol{u} = k\boldsymbol{u} + c\boldsymbol{u}$ (*distributive law*)
7. $(kc)\boldsymbol{u} = k(c\boldsymbol{u})$ (*associative law*)
8. 모든 벡터 $\boldsymbol{u}$에 대해서 $1\boldsymbol{u} = \boldsymbol{u}$ (*neutral element*)를 가진다.

위의 Proposition 2은 Matrix properties를 따른다.

Proposition 3. $\boldsymbol{u}$가 R^n에 있는 어떤 벡터라 하자. 그러면 $\boldsymbol{u} + (-\boldsymbol{u}) = \boldsymbol{O}$를 만족시키는 벡터 $-\boldsymbol{u}$는 유일(*Unique*)하다.

위의 Proposition 3를 증명해보자.

Proof 7 벡터 $\mathbf{v}$가 다음을 만족시키는 R^n에 있는 어떤 벡터라 하자.

$$\mathbf{u} + \mathbf{v} = \mathbf{O} \tag{35}$$

$\mathbf{v} = -\mathbf{u}$를 보이면 이에 대한 증명을 할 수 있다.

식 (35)의 양변에 $-\mathbf{u}$를 더하면 다음과 같다.

$$
\begin{aligned}
-\mathbf{u} + (\mathbf{u} + \mathbf{v}) &= (-\mathbf{u}) + \mathbf{O} \\
(-\mathbf{u} + \mathbf{u}) + \mathbf{v} &= (-\mathbf{u}) + \mathbf{O} \\
(\mathbf{u} + (-\mathbf{u})) + \mathbf{v} &= -\mathbf{u} \\
\mathbf{O} + \mathbf{v} &= -\mathbf{u} \\
\mathbf{v} &= -\mathbf{u} \qquad \square
\end{aligned} \tag{36}
$$

이는 R^n에 있는 모든 벡터 $\mathbf{u}$에 대해서 R^n에서 다음을 만족시키는 벡터는 오직 $-\mathbf{u}$ 외에는 없다는 의미이다.

$$
\mathbf{u} + (-\mathbf{u}) = \mathbf{O} \tag{37}
$$

이번에는 다음의 명제(Proposition)를 증명해보자.

Proposition 4. *$\boldsymbol{u}$가 R^n에 있는 어떤 벡터라 하자. 그러면 다음을 만족시킨다.*

$$
(-1)\boldsymbol{u} = -\boldsymbol{u} \tag{38}
$$

Proof 8 $\mathbf{u} = \begin{pmatrix} u_1 \\ \vdots \\ u_n \end{pmatrix}$이라 할 때, 먼저 $\mathbf{u} + (-1)\mathbf{u} = \mathbf{O}$를 보인 다음, 위 Proposition 3를 이용한다.

$$
\mathbf{u} + (-1)\mathbf{u} = \begin{pmatrix} u_1 \\ \vdots \\ u_n \end{pmatrix} + (-1) \begin{pmatrix} u_1 \\ \vdots \\ u_n \end{pmatrix} = \begin{pmatrix} u_1 \\ \vdots \\ u_n \end{pmatrix} + \begin{pmatrix} -u_1 \\ \vdots \\ -u_n \end{pmatrix} \tag{39}
$$

$$
= \begin{pmatrix} u_1 - u_1 \\ \vdots \\ u_n - u_n \end{pmatrix} = \begin{pmatrix} 0 \\ \vdots \\ 0 \end{pmatrix} = \mathbf{O} \tag{40}
$$

이로부터 $\mathbf{u} + (-1)\mathbf{u} = \mathbf{O}$를 만족시킴을 확인하였고, 여기서 $-\mathbf{u}$는 Proposition 3에 의하여 오직 하나임을 알 수 있다.

$$\therefore (-1)\mathbf{u} = -\mathbf{u} \qquad \square \tag{41}$$

—***Jonghyeok Kim***

2.1.2 Dot(inner) product revisited

R^n에 있는 $\mathbf{u} = \begin{pmatrix} u_1 \\ \vdots \\ u_n \end{pmatrix}$와 $\mathbf{v} = \begin{pmatrix} v_1 \\ \vdots \\ v_n \end{pmatrix}$로 정의된다고 하자. 그러면 $\mathbf{u}$와 $\mathbf{v}$의 Dot product는 $\mathbf{u}\cdot\mathbf{v}$로 표기되고, 다음과 같이 정의된다.

$$\mathbf{u}\cdot\mathbf{v} = u_1v_1 + u_2v_2 + \cdots + u_nv_n \tag{42}$$

이는 Column vector $\mathbf{v}$와 $\mathbf{u}$의 Transpose의 Matrix 곱셈과 동일하다.

즉,

$$\mathbf{u}\cdot\mathbf{v} = \mathbf{u}^T\mathbf{v} = \begin{pmatrix} u_1 \\ \vdots \\ u_n \end{pmatrix}^T \begin{pmatrix} v_1 \\ \vdots \\ v_n \end{pmatrix} \tag{43}$$

$$= \begin{pmatrix} u_1 & u_2 & \cdots & u_n \end{pmatrix} \begin{pmatrix} v_1 \\ \vdots \\ v_n \end{pmatrix} = u_1v_1 + u_2v_2 + \cdots + u_nv_n \tag{44}$$

만약 어떤 벡터가 하나의 열만을 가지고 있는 Matrix라면, Matrix 곱셈이 오직 좌측에 있는 Matrix $\mathbf{u}$의 열의 수와 우측에 있는 Matrix $\mathbf{v}$의 행의 수가 일치할 때에만 성립하기 때문에 좌측에 있는 벡터 $\mathbf{u}$를 Transpose하는 것이다.

이러한 Matrix 곱셈을 벡터들 $\mathbf{u}$와 $\mathbf{v}$의 Dot 또는 Inner product라 하고, 다음과 같이 연산된다.

$$\mathbf{u}\cdot\mathbf{v} = \mathbf{u}^T\mathbf{v} = u_1v_1 + u_2v_2 + \cdots + u_nv_n \tag{45}$$

위의 식 (45)의 결과를 보면, R^n에 있는 두 개의 벡터들의 Dot product의 결과는 실수(Real number)임을 알 수 있다. 즉, 두 벡터들에 대한 Dot product의 결과는 벡터가 아닌 스칼라이다.

또한, 두 벡터들 $\mathbf{u}$와 $\mathbf{v}$의 Dot product는 각 Component u_j와 그에 상응하는 Component인 v_j를 각각 곱한 후, 그 결과들을 모두 더함으로써 구할 수 있다.

만약 다음과 같은 방정식이 주어졌을 때,

$$3x + 4y = 11 \tag{46}$$

Dot product를 사용하여 이를 다음과 같이 표현할 수 있다.

$$\begin{pmatrix} 3 \\ 4 \end{pmatrix} \cdot \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} 3 & 4 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = 11 \tag{47}$$

벡터 $\mathbf{u}$와 $\mathbf{v}$가 Perpendicular 또는 Orthogonal하다는 그들의 Dot product의 결과가 0인 것과 동치이다.

즉, R^n에 있는 두 벡터 $\mathbf{u}$와 $\mathbf{v}$가 Perpendicular 또는 Orthogonal하다.

$$\iff \mathbf{u} \cdot \mathbf{v} = 0 \tag{48}$$

특히, 선형대수학에서 Orthogonality의 개념은 매우 중요하다.

위의 식 (48)를 $3x + 4y = 0$의 방정식을 해결하는 데 사용할 수도 있다.

즉 다음의 식을 만족시키는 모든 벡터 $\begin{pmatrix} x & y \end{pmatrix}^T$를 찾으면 되는 것이다.

$$\begin{pmatrix} 3 \\ 4 \end{pmatrix} \cdot \begin{pmatrix} x \\ y \end{pmatrix} = 0 \tag{49}$$

그런데 이미 앞서 살펴본 것처럼, 만약 두 벡터 $\mathbf{u}$와 $\mathbf{v}$의 Dot product $\mathbf{u} \cdot \mathbf{v}$의 결과가 0이라면, $\mathbf{u}$와 $\mathbf{v}$d는 서로 Perpendicular하다.

이러한 개념을 이용하여 벡터 $\begin{pmatrix} 3 \\ 4 \end{pmatrix}$에 Perpendicular (Orthogonal)한 벡터들을 찾으면 된다.

특히, 벡터 $\begin{pmatrix} 3 \\ 4 \end{pmatrix}$는 R^2을 형성하므로, 즉 $n \leq 3$이므로, 기하적 해석이 가능하고, 그러므로 이를 그림으로 나타내면 다음과 같다.

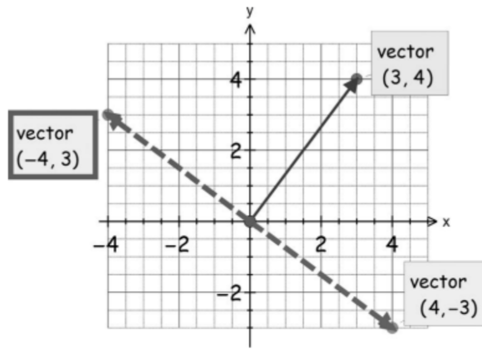

Figure 3: The perpendicular (orthogonal) vectors to the vector $\begin{pmatrix} 3 \\ 4 \end{pmatrix}$

위 Figure 3에서 확인할 수 있듯이, Dashed line 상에 있는 모든 벡터는 벡터 $\begin{pmatrix} 3 \\ 4 \end{pmatrix}$에 수직인 것을 확인할 수 있을 뿐만 아니라, 주어진 방정식 $3x + 4y = 0$은 An infinite number of solutions을 가짐도 확인해볼 수 있다.

—***Jonghyeok Kim***

2.1.3 Angle between two vectors

두 Zero가 아닌 Vectors들 사이의 각도에 대한 공식은 2-space에서 N-space까지 확장될 수 있는데, 즉 **u**와 **v**를 R^n에 있는 어떤 Zero가 아닌 Vectors들이라 하자. 그러면 다음이 성립한다.

$$cos(\theta) = \frac{\mathbf{u} \cdot \mathbf{v}}{||\mathbf{u}||||\mathbf{v}||} \qquad \text{where } 0 \leq \theta \leq \pi \ (radians) \tag{50}$$

—***Jonghyeok Kim***

2.1.4 Inequalities

다음으로, Vectors들의 Dot product와 Norm(Length)와 관련된 몇 가지 부등식(Inequalities)들을 증명해 보자.

—***Jonghyeok Kim***

2.1.5 Properties of linear dependence

Proposition 5. $\boldsymbol{v}_1, \boldsymbol{v}_2, \cdots \boldsymbol{v}_n$이 R^n에 있는 벡터들이라 하자. 만약 이러한 벡터들 중 최소 하나의 벡터 가령 벡터 $\boldsymbol{v}_j$가 *Zero* 벡터라면, $\boldsymbol{v}_1, \boldsymbol{v}_2 \cdots \boldsymbol{v}_n$은 *Linearly dependent*하다.

위의 Proposition 5를 증명해보자.

Proof 9 다음의 Linear combination을 고려해보자.

$$k_1\mathbf{v}_1 + k_2\mathbf{v}_2 + \cdots + k_j\mathbf{v}_j + \cdots + k_n\mathbf{v}_n = \mathbf{O} \tag{51}$$

위의 식 (51)에서 $k_j \neq 0$ (Non-zero number)를 취한 다음, 다른 모든 Scalars들은 0과 같다고 해보자. 즉, 다음과 같다고 해보자.

$$k_1 = k_2 = \cdots = k_{j-1} = k_{j+1} = \cdots = k_n = 0 \tag{52}$$

$\mathbf{v}_j = O$이기 때문에 $k_j\mathbf{v}_j = O$이고, 이는 (51)에서 모든 Scalars들은 0이 아님($k_j \neq 0$)을 의미한다.

결론적으로, 만약 Zero가 아닌 k가 $k_1\mathbf{v}_1 + k_2\mathbf{v}_2 + \cdots + k_n\mathbf{v}_n = \mathbf{O}$를 만족시킨다면, 벡터 $\mathbf{v}$는 Dependent 하다.

정리하면, $k_j \neq 0$이기 때문에 벡터 $\mathbf{v}_1, \mathbf{v}_2, \cdots, \mathbf{v}_n$은 Linearly dependent하다. □

위 Proposition 5이 의미하는 바는 만약 벡터들 $\mathbf{v}_1, \mathbf{v}_2, \cdots, \mathbf{v}_n$ 중 하나가 Zero 벡터라면 이러한 벡터들은 Linearly dependent함을 의미한다.

Proposition 6. $\boldsymbol{v}_1, \boldsymbol{v}_2, \cdots, \boldsymbol{v}_m$이 R^n 내의 서로다른 벡터들이라 하자.

만약 $n < m$이라면, 즉 *n-space*에서 n의 값이 벡터들의 수 m보다 작다면, 벡터들 $\boldsymbol{v}_1, \boldsymbol{v}_2, \cdots, \boldsymbol{v}_m$은 *Linearly dependent*하다.

위 Proposition 6를 증명해보자.

Proof 10 주어진 벡터들 $\mathbf{v}_1, \mathbf{v}_2, \cdots, \mathbf{v}_m$의 다음 Linear combination을 고려해보자.

$\mathrm{k}_1\mathbf{v}_1 + k_2\mathbf{v}_2 + \cdots + k_n\mathbf{v}_n + k_{n+1}\mathbf{v}_{n+1} + \cdots + k_m\mathbf{v}_m = \mathbf{O}$

각 벡터가 n-space R^n에 속하기 때문에 방정식들의 수는 n이고, 반면 미지수 $k_1, k_2, \cdots, k_n, k_{n+1}, \cdots, k_m$의 수는 m이다. 식 (10)은 다음과 같이 작성할 수 있다.

$$\underbrace{k_1\begin{pmatrix}v_{11}\\v_{12}\\\vdots\\v_{1n}\end{pmatrix}+k_2\begin{pmatrix}v_{21}\\v_{22}\\\vdots\\v_{2n}\end{pmatrix}+\cdots+k_n\begin{pmatrix}v_{n1}\\v_{n2}\\\vdots\\v_{nn}\end{pmatrix}+k_{n+1}\begin{pmatrix}v_{(n+1)1}\\v_{(n+1)2}\\\vdots\\v_{(n+1)n}\end{pmatrix}+\cdots+k_m\begin{pmatrix}v_{m1}\\v_{m2}\\\vdots\\v_{mn}\end{pmatrix}}_{m\text{ unknowns}}$$

$$=\left.\begin{pmatrix}0\\0\\\vdots\\0\end{pmatrix}\right\}n\text{ equations}$$

—***Jonghyeok Kim***

2.1.6 Basis

Definition 2. *n space* R^n에 있는 n개의 벡터들 $\boldsymbol{v}_1, \boldsymbol{v}_2, \cdots, \boldsymbol{v}_n$을 고려해보자.

이러한 벡터들이 R^n에 대한 *Basis*를 형성한다. $\iff$

- $\boldsymbol{v}_1, \boldsymbol{v}_2, \cdots, \boldsymbol{v}_n$은 R^n을 *Span*하고,
- $\boldsymbol{v}_1, \boldsymbol{v}_2, \cdots, \boldsymbol{v}_n$은 *Linearly independent*하다.

—***Jonghyeok Kim***

2.1.7 Properties of bases

Proposition 7. 어떤 R^n에 있는 n개의 *Linearly independent*한 벡터들은 R^n에 대한 *Basis*를 형성한다.

즉, 위의 Proposition 7은 중요한 결과인데, 그 이유인 즉슨 이것이 n-space R^n에 있는 n개의 벡터들이 주어진다면, 그것들이 Linearly independent함만을 보이면 Basis를 형성한다는 것이다. 다시 말해서, R^n을 Span함을 보일 필요가 없다.

예를 들어, R^3에 있는 3개의 Linearly independent한 벡터들은 R^3에 대한 Basis를 형성한다. 마찬가지로, R^{103}에 있는 103개의 Linearly independent한 벡터들은 R^{103}에 대한 Basis를 형성한다.

Proposition 8. R^n을 *Span*하는 n개의 벡터들은 R^n에 대한 *Basis*를 형성한다.

이번에도 위의 Proposition 8에 따라 n개의 벡터들이 R^n을 Span함을 보이면, R^n에 대한 Basis임이 보장된다.

예를 들어, R^{666}을 Span하는 666개의 벡터들이 있다면, 이러한 벡터들이 R^{666}에 대한 Basis를 형성한다.

그러므로 Proposition 7와 8로부터 R^n에 있는 n개의 벡터들을 가진다면, 이들이 Independence한지 Span한지 둘 중에 하나의 조건만을 만족시킨다면, 이들 벡터들이 R^n에 대한 Basis를 형성한다.

—***Jonghyeok Kim***

3 Conclusion

3.1 Conclusion

4 Acknowledgement

Appendix A

A Types of Matrices

A.1 Tutorial Overview

Matrices는 크게 다음과 같이 6가지 종류로 나뉘어진다.

1. Square Matrix
2. Symmetric Matrix
3. Triangular Matrix
4. Diagonal Matrix
5. Identity Matrix
6. Orthogonal Matrix

위의 Matrix 중 4번을 제외한 모든 Matrix는 1번의 Square Matrix에 포함된다. (즉, Square Matrix이다.)

A.1.1 Square Matrix

Square Matrix는 행(Rows) n의 수와 열(Columns) m의 수가 동일한 Matrix이다.

$$n \equiv m \tag{1}$$

이러한 Square Matrix는 행(Rows)과 열(Columns)의 수가 동일하지 않은 Rectangular Matrix와는 대조된다.

Square Matrix의 차원은 보통 n을 이용하여 표기되는 데, 예를 들면 다음과 같은 형식으로 표기된다.

$$n \times n \tag{2}$$

또한, Matrix의 크기를 Order라고 하는 데, 예를 들어, Order 4의 Square Matrix는 4×4로 표기된다.

즉, Square Matrix는 행(Rows)과 열(Columns)이 동일하기 때문에 Matrix의 크기 Order 4가 하나만 주어지면 나머지 하나의 정보를 알 수 있다.

특히, Square Matrix의 Top left에서 Bottom right까지의 Diagonal상의 값들에 대한 Vector를 Main diagonal이라 부른다.

예를 들어, 아래와 같은 Square Matrix가 있다고 하면,

$$\begin{pmatrix} 1 & 2 & 3 \\ 1 & 2 & 3 \\ 1 & 2 & 3 \end{pmatrix} \tag{3}$$

이때 Main diagonal(Main diagonal은 정의에서 이미 확인한 것처럼, Vector이다.)은 다음과 같다.

$$\begin{pmatrix} 1 & 2 & 3 \end{pmatrix} \tag{4}$$

이러한 Square Matrix는 서로 손쉽게 더해지고 곱해질 수 있으며, 회전(Rotations)(e.g. 이미지 회전)과 같은 많은 단순한 선형 변환(Linear transformation)의 기초(Basis)가 된다.

A.1.2 Symmetric Matrix

Symmetric Matrix는 Square Matrix의 한 종류로서, Top-right Triangle과 Bottom-left Triangle이 동일한 Matrix이다.

Introduction to Linear Algebra Fifth Edition(2016)의 Symmetric Matrices에 대한 다음의 설명을 참조해보자.

> “It is no exaggeration to say that symmetric matrices S are the most important matrices the world will ever see - in the theory of linear algebra and also in the applications.”

직역해보면, 선형대수학의 이론과 실제 응용에 있어서 전세계를 통틀어 Symmetric Matrices가 가장 중요한 Matrices라고 말하는 것은 과장되지 않았다는 의미이다. 즉, 그만큼 Symmetric Matrices가 중요한 Matrices 라는 의미이다.

어떤 Matrices가 Symmetric이 되기 위해서는 대칭축(The axis of symmetry)이 항상 그 Matrix의 Main diagonal (Top left $\sim$ Bottom right)이어야 한다.

즉, Main diagonal이라는 용어는 Square Matrices에서 정의되므로, 결국 Symmetric Matrices는 이러한 Square Matrices의 한 종류임을 유추해볼 수 있다.

이러한 Symmetric Matrices의 예를 보이면 다음과 같다.

$$\begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 2 & 1 & 2 & 3 & 4 \\ 3 & 2 & 1 & 2 & 3 \\ 4 & 3 & 2 & 1 & 2 \\ 5 & 4 & 3 & 2 & 1 \end{pmatrix} \tag{5}$$

위의 Matrix (5)를 보면, Main diagonal 기준(대칭축)으로 그 Entries들이 서로 동일함을 확인해볼 수 있다. 즉, 해당 Matrix는 Symmetric Matrix이다.

어떠한 Symmetric Matrix가 있다면, 그것은 항상 Square하고 그 자신의 Transpose와 동일하다.

여기서 Transpose란 행과 열의 수를 뒤바꾸는 즉 그 위치를 서로 뒤바꾸는 연산(Operations)이다.

$$M = M^T \tag{6}$$

A.1.3 Triangular Matrix

Triangular Matrix란 Square Matrix의 한 종류인데, 그 Matrix의 Upper-right 또는 Lower-left에서의 모든 값들이 0의 값으로 채워진 Matrix이다.

Main diagonal의 위쪽에만 값(0이 아닌 값)이 있는 Triangular Matrix를 Upper-triangular Matrix라 하고, 반대로 Main diagonal의 아랫쪽에만 값(0이 아닌 값)이 있는 Triangular Matrix를 Lower-triangular Matrix라 한다.

즉, 이러한 Triangular Matrix 역시 그 정의상에서 Square Matrix의 정의에 등장하는 Main diagonal이라는 개념이 수반되었으므로, 이 역시 Square Matrix임을 유추해볼 수 있다.

아래는 3×3의 Upper-triangular Matrix의 한 예제이다.

$$\begin{pmatrix} 1 & 2 & 3 \\ 0 & 2 & 3 \\ 0 & 0 & 3 \end{pmatrix} \tag{7}$$

아래는 3×3의 Lower-triangular Matrix의 한 예제이다.

$$
\begin{pmatrix} 1 & 0 & 0 \\ 1 & 2 & 0 \\ 1 & 2 & 3 \end{pmatrix} \tag{8}
$$

위의 Upper-triangular Matrix (7)는 Linear Systems을 해결함에 있어 Gaussian elimination with back substitute 을 적용할 때, 최종 Matrix의 모습임을 알 수 있다.

다시 말해서, 만약에 다음과 같은 Linear Systems가 있다고 해보자.

$$
\begin{cases} x + 2y + 3z = 0 \\ 2x + 6y + 9z = 0 \\ 3x + 6y + 12z = 0 \end{cases} \tag{9}
$$

위의 식 (9)을 Augmented Matrix로 작성하면 다음과 같다.

$$
\left(\begin{array}{ccc|c} 1 & 2 & 3 & 0 \\ 2 & 6 & 9 & 0 \\ 3 & 6 & 12 & 0 \end{array}\right) \tag{10}
$$

위의 행렬 (11)에 Gaussian elimination을 적용하게 되면 다음과 같고,

$$
\left(\begin{array}{ccc|c} 1 & 2 & 3 & 0 \\ 0 & 2 & 3 & 0 \\ 0 & 0 & 3 & 0 \end{array}\right) \tag{11}
$$

이는 위에서 살펴본 Upper-triangular Matrix (7)와 동일하고(이 경우에 한해서만), Upper-triangular Matrix 형태임을 확인해볼 수 있다. (중요한 것은 Upper-triangular Matrix 형태라는 사실이다.)

결론적으로, Upper-triangular인 Augmented Matrix (11)을 Linear Systems로 변환하면 다음과 같고,

$$
\begin{cases} x + 2y + 3z = 0 \\ 2y + 3z = 0 \\ 3z = 0 \end{cases} \tag{12}
$$

마지막으로, Back substitute를 적용하게 되면 다음과 같다. (참고로, Back substitute란 용어는 필자 생각에

Linear systems를 Gaussian elimination을 적용하여 Upper-triagular Matrix를 도출하는 방향은 위에서 아래이지만, Substitute(치환)는 아래에서 위 방향으로 거슬러 올라가기 때문에 Back이란 단어가 접두사로 붙어 Back substitute라 말하는 것으로 이해하였다.)

$$\begin{cases} x + 2y + 3z = 0 \ (x = 0; \ \because y = 0, z = 0) \\ \quad 2y + 3z = 0 \ (y = 0, z = 0; \ \because z = 0) \\ \qquad\qquad 3z = 0 \ (z = 0) \end{cases} \tag{13}$$

다시 본론으로 돌아와서, Numpy에는 어떤 Square Matrix를 Triangular Matrix로 변환해주는 함수를 제공하는데, **tril**와 **triu** 함수들이 바로 그것이다.

tril 함수는 인자로 전달되는 Square Matrix를 Lower-triangular Matrix로 변환해주고, **triu** 함수는 인자로 전달되는 Square Matrix를 Upper-triangular Matrix로 변환해준다.

다음 예제는 우선 3×3의 Square Matrix를 정의하고, 그것을 Lower-triangular와 Upper-triangular Matrix로 변환하는 과정이 수록되었다.

```
from numpy import array
from numpy import tril
from numpy import triu
# Square Matrix M 정의
M = array([
    [1, 2, 3],
    [1, 2, 3],
    [1, 2, 3]])
print(M)
# Lower-triangular Matrix
lower = tril(M)
print(lower)
# Upper-triangular Matrix
upper = triu(M)
print(upper)
```

B References

[1] Jonghyeok Kim. *Is it wrong sentence*? 2022. URL: `https://math.stackexchange.com/questions/4614264/is-it-wrong-sentence` (visited on 01/09/2023).

[2] Kuldeep Singh. *Linear Algebra*: *Step by Step*. Oxford University Press, 2013.