CHAPTER 03

LINEAR ALGEBRA

행렬식

개요

- 행렬식과 관련된 전반적인 논제들을 학습
- 행렬식의 기본적인 개념과 여인수에 의한 행렬식을 계산함
- 행렬식에서 기본 행 연산을 통한 행렬식의 계산법을 살펴봄
- 역행렬의 정의와 성질을 고찰하고, 가우스-조단 방법과 수반행렬에 의한 역행렬을 구하는 방법을 탐구함
- 역행렬과 크래머의 규칙을 이용한 선형방정식의 해를 구하는 방법을 고찰하고 행렬식을 이용한 응용도 살펴봄
- C 프로그램과 MATLAB을 통하여 본문에 있는 행렬식, 역행렬, 크래머의 규칙 등의 결과를 비교하며 실습해 보고 학습함

03

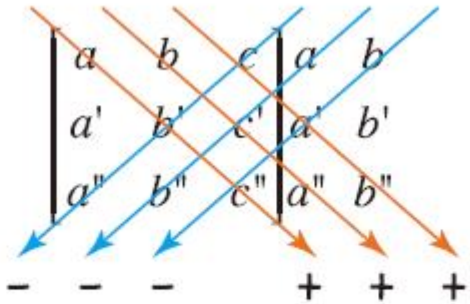
CHAPTER

행렬식

LINEAR ALGEBRA

CONTENTS

5



$$\begin{vmatrix} x^2 & x & y & 1 \\ 9 & 3 & 5 & 1 \\ 4 & -2 & 3 & 1 \\ 36 & 6 & 9 & 1 \end{vmatrix} = 0$$

- **행렬식(Determinant, 行列式)**이란 정방행렬 A에 하나의 스칼라 값을 대응시키는 함수로써 보통 Det(A) 또는 |A|로 표시한다. n차 정방행렬의 행렬식을 n차 행렬식이라고도 부른다.

여기서 잠깐 !!

행렬식이란 용어는 가우스(Gauss)에 의해 처음으로 소개되었는데, 그는 행렬식이 행렬의 성질을 결정할 수 있다고 믿었다. 그 후 코시(Cauchy)에 의해 현실적 개념의 행렬식이 쓰여졌다. 행렬의 개념은 우리의 짐작과는 달리 행렬식의 개념이 소개된 지 무려 150년이 지난 후에야 널리 알려졌다.

정의 ❸-1 1×1 행렬 $A=[a_{11}]$의 행렬식은 다음과 같이 정의된다.

$$\text{Det}(A)=|a_{11}|=a_{11}$$

2×2 행렬 $A=\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}$의 행렬식은 다음과 같이 정의된다.

$$\text{Det}(A)=\begin{vmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{vmatrix}=a_{11}a_{22}-a_{12}a_{21}$$

따라서 1×1 행렬 $[a]$의 행렬식은 $|a|$로써 행렬식의 값은 a이고,

2×2 행렬 $\begin{bmatrix} a & b \\ c & d \end{bmatrix}$의 행렬식은 $\begin{vmatrix} a & b \\ c & d \end{vmatrix}$로써 행렬식의 값은 $ad-bc$이다.

3.1 행렬식의 개념과 여인수

정의 ❸-2 3×3 행렬 A의 행렬식은 다음과 같이 정의된다.

$$A=\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}$$ 의 행렬식은 다음과 같다.

$$\mathrm{Det}(A)=\begin{vmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{vmatrix}=a_{11}a_{22}a_{33}+a_{12}a_{23}a_{31}+a_{13}a_{21}a_{32} -a_{11}a_{23}a_{32}-a_{12}a_{21}a_{33}-a_{13}a_{22}a_{31}$$

3.1 행렬식의 개념과 여인수

C program, MATLAB

예제 ❸-1

다음 행렬 A, B의 행렬식을 각각 구해 보자.

$$A=\begin{bmatrix}2 & 1\\ 1 & 4\end{bmatrix},\quad B=\begin{bmatrix}-2 & -3\\ 4 & 5\end{bmatrix}$$

풀이 A의 행렬식은 $2\cdot 4-1\cdot 1=7$이다.

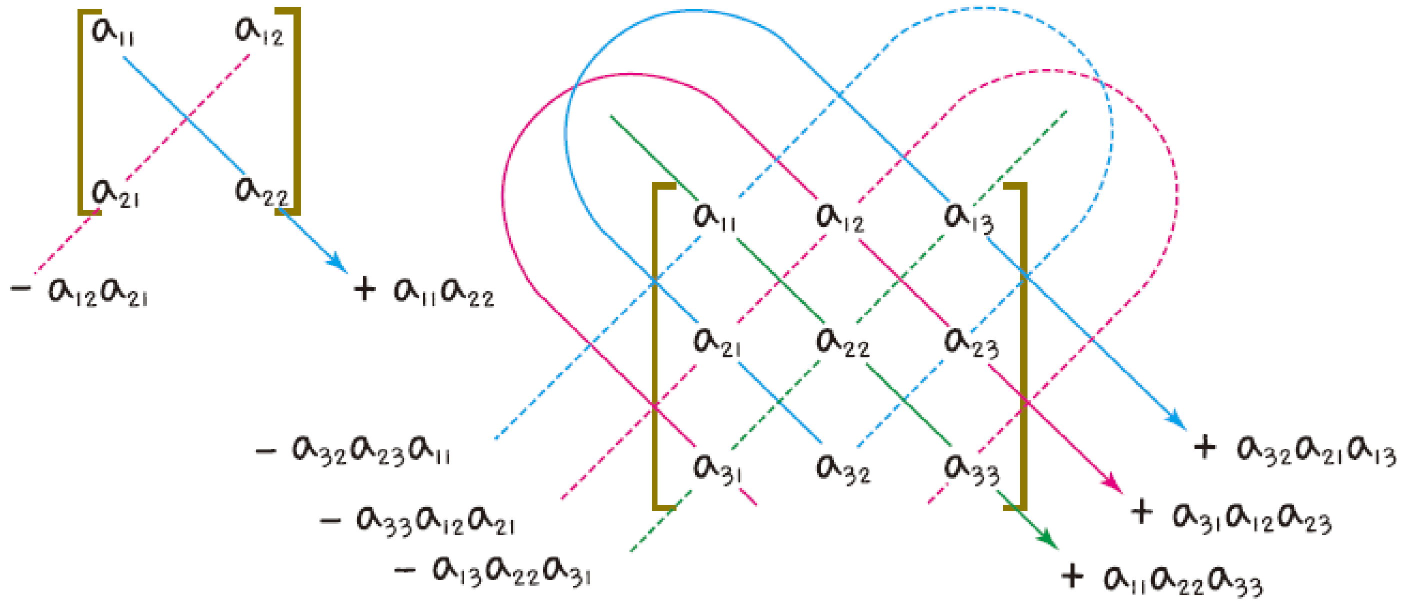
또한 B의 행렬식은 $(-2)\cdot 5-(-3)\cdot 4=-10+12=2$이다. ■

예제 ❸-2 행렬 $A=\begin{bmatrix}1&2&3\\2&1&3\\3&1&2\end{bmatrix}$가 주어졌을 때 $|A|$를 구해 보자.

풀이 3×3 행렬의 행렬식을 구하는 공식에 따라 계산하면

$$|A|=\begin{vmatrix}1&2&3\\2&1&3\\3&1&2\end{vmatrix}=1\cdot1\cdot2+2\cdot3\cdot3+3\cdot2\cdot1 \\ -1\cdot3\cdot1-2\cdot2\cdot2-3\cdot1\cdot3=6 \quad ■$$

또한 우리는 〈그림 3.1〉과 같은 사루스의 공식(Sarrus's Formula)에 따라 행렬식을 구할 수도 있는데, 그 결과는 위의 방법과 같다. 2×2 행렬식이나 3×3 행렬식의 경우 도식적인 방법을 쓰면 계산이 더 편리한데, 화살표가 지나는 문자를 곱한 것에다 화살표 끝의 부호를 붙여서 합하면 된다.



a_{11} a_{12}
a_{21} a_{22}
$-a_{12}a_{21}$
$+a_{11}a_{22}$
a_{11} a_{12} a_{13}
a_{21} a_{22} a_{23}
a_{31} a_{32} a_{33}
$-a_{32}a_{23}a_{11}$
$-a_{33}a_{12}a_{21}$
$-a_{13}a_{22}a_{31}$
$+a_{32}a_{21}a_{13}$
$+a_{31}a_{12}a_{23}$
$+a_{11}a_{22}a_{33}$

〈그림 3.1〉 사루스의 공식

3.1.2 여인수에 의한 행렬식의 계산

행렬식을 계산하는 더 일반적인 방법을 살펴보자. 여기서는 편의상 3×3 행렬식에 관한 것부터 고찰한다. 위의 사루스의 공식에 따라

$$\mathrm{Det}(A) = \begin{vmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{vmatrix} = a_{11}a_{22}a_{33} - a_{11}a_{23}a_{32} - a_{12}a_{21}a_{33} + a_{12}a_{23}a_{31} + a_{13}a_{21}a_{32} - a_{13}a_{22}a_{31} \quad (1)$$

(1)의 식을 더욱 쉽게 다루기 위해 공통된 항을 묶어서 인수분해를 하면 다음과 같다.

$$\mathrm{Det}(A) = a_{11}(a_{22}a_{33} - a_{23}a_{32}) + a_{12}(-a_{21}a_{33} + a_{23}a_{31}) + a_{13}(a_{21}a_{32} - a_{22}a_{31})$$

이것은 다음 행렬식들에서 a_{11}, a_{12}, a_{13}을 기준으로 해서, 해당하는 행과 열을 제외하고 연산을 하는 것과 같은 결과를 가져온다.

$$a_{11}\begin{vmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{vmatrix} - a_{12}\begin{vmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{vmatrix} + a_{13}\begin{vmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{vmatrix}$$

여기서 우리는 a_{ij}를 포함하는 행과 열을 제외한 나머지 행과 열로 이루어진 $(n-1)\times(n-1)$ 행렬식 M_{ij}를 a_{ij}의 **소행렬식(minor)**으로 정의한다. 소행렬식 M_{ij}는 A의 제i행과 제j열을 제거하여 얻어진 부분행렬의 행렬식이다. 따라서 M_{11}은 A의 제1행과 제1열을 제거하여 얻어진 부분행렬식이고, M_{12}는 A의 제1행과 제2열을 제거하여 얻은 부분행렬식이며, M_{13}은 A의 제1행과 제3열을 제거하여 얻은 부분행렬식이다. 즉,

$$M_{11} = \begin{vmatrix} a_{22} & a_{23} \\ a_{32} & a_{33} \end{vmatrix},\quad M_{12} = \begin{vmatrix} a_{21} & a_{23} \\ a_{31} & a_{33} \end{vmatrix},\quad M_{13} = \begin{vmatrix} a_{21} & a_{22} \\ a_{31} & a_{32} \end{vmatrix}$$ 이다.

따라서 (2)식은 다음과 같이 표현될 수 있다.

$$\mathrm{Det}(A) = a_{11}M_{11} - a_{12}M_{12} + a_{13}M_{13} \tag{3}$$

M_{ij}에다 $i+j$의 값에 따라 부호를 넣은 A_{ij}를 **여인수(cofactor, 餘因數)** 또는 **여인자(餘因子)**라고 하는데

$$A_{ij} = (-1)^{i+j}M_{ij}$$

를 적용하면 3개의 여인수는 각각

$$A_{11} = \begin{vmatrix} a_{22} & a_{23} \\ a_{32} & a_{33} \end{vmatrix} \qquad A_{12} = -\begin{vmatrix} a_{21} & a_{23} \\ a_{31} & a_{33} \end{vmatrix}$$

$$A_{13} = \begin{vmatrix} a_{21} & a_{22} \\ a_{31} & a_{32} \end{vmatrix}$$

과 같이 된다. 따라서 (3)식은

$$\mathrm{Det}(A) = a_{11}A_{11} + a_{12}A_{12} + a_{13}A_{13} \tag{4}$$

와 같이 된다.

(4)식을 제1행에 대해 **여인수들로 전개되었다**라고 말한다.

3.1 행렬식의 개념과 여인수

여인수는 소행렬식에다 부호를 붙인 것인데, $i+j$가 짝수이면 $A_{ij}=M_{ij}$이고, $i+j$가 홀수이면 $A_{ij}=-M_{ij}$이다.

3×3 행렬은 다음과 같이 모두 9개의 여인수를 가진다.

$$\begin{array}{lll} A_{11}=\ \ M_{11} & A_{12}=-M_{12} & A_{13}=\ \ M_{13} \\ A_{21}=-M_{21} & A_{22}=\ \ M_{22} & A_{23}=-M_{23} \\ A_{31}=\ \ M_{31} & A_{32}=-M_{32} & A_{33}=\ \ M_{33} \end{array}$$

여인수에 대응하는 부호는 3×3 행렬인 경우 (5)와 같이 행과 열의 합의 값에 따라 교대로 부호가 바뀐다.

$$\begin{array}{ccc} + & - & + \\ - & + & - \\ + & - & + \end{array} \tag{5}$$

4×4 행렬인 경우에는 (6)과 같이 확장될 수 있다.

$$\begin{array}{cccc} + & - & + & - \\ - & + & - & + \\ + & - & + & - \\ - & + & - & + \end{array} \tag{6}$$

여기서 잠깐!!

행렬식을 구할 때 사루스 공식에 의한 방법과 여인수에 의한 방법의 결과와 차이점은 다음과 같다. 사루스 공식에 의한 방법은 행렬의 크기가 3×3 이하인 경우에만 계산이 가능하고, 그보다 큰 행렬인 경우에는 사루스의 공식을 적용할 수 없으므로 여인수에 의한 방법을 사용해야 한다. 하지만 두 방법의 결과는 같다.

C program, MATLAB

예제 ❸-3

다음 행렬 A의 행렬식을 제1행에 대한 여인수들로 전개하여 구해 보자.

$$A=\begin{bmatrix}2&4&7\\6&0&3\\1&5&3\end{bmatrix}$$

풀이 제1행에 대해 여인수들로 전개하면 다음과 같다.

$$\mathrm{Det}(A)=\begin{vmatrix}2&4&7\\6&0&3\\1&5&3\end{vmatrix}=2A_{11}+4A_{12}+7A_{13}$$

$$\begin{vmatrix}2&4&7\\6&0&3\\1&5&3\end{vmatrix}\qquad\begin{vmatrix}2&4&7\\6&0&3\\1&5&3\end{vmatrix}\qquad\begin{vmatrix}2&4&7\\6&0&3\\1&5&3\end{vmatrix}$$

이제 A의 제1행에 있는 원소들의 여인수들은 다음과 같다.

$$A_{11} = (-1)^{1+1} \begin{vmatrix} 0 & 3 \\ 5 & 3 \end{vmatrix}$$

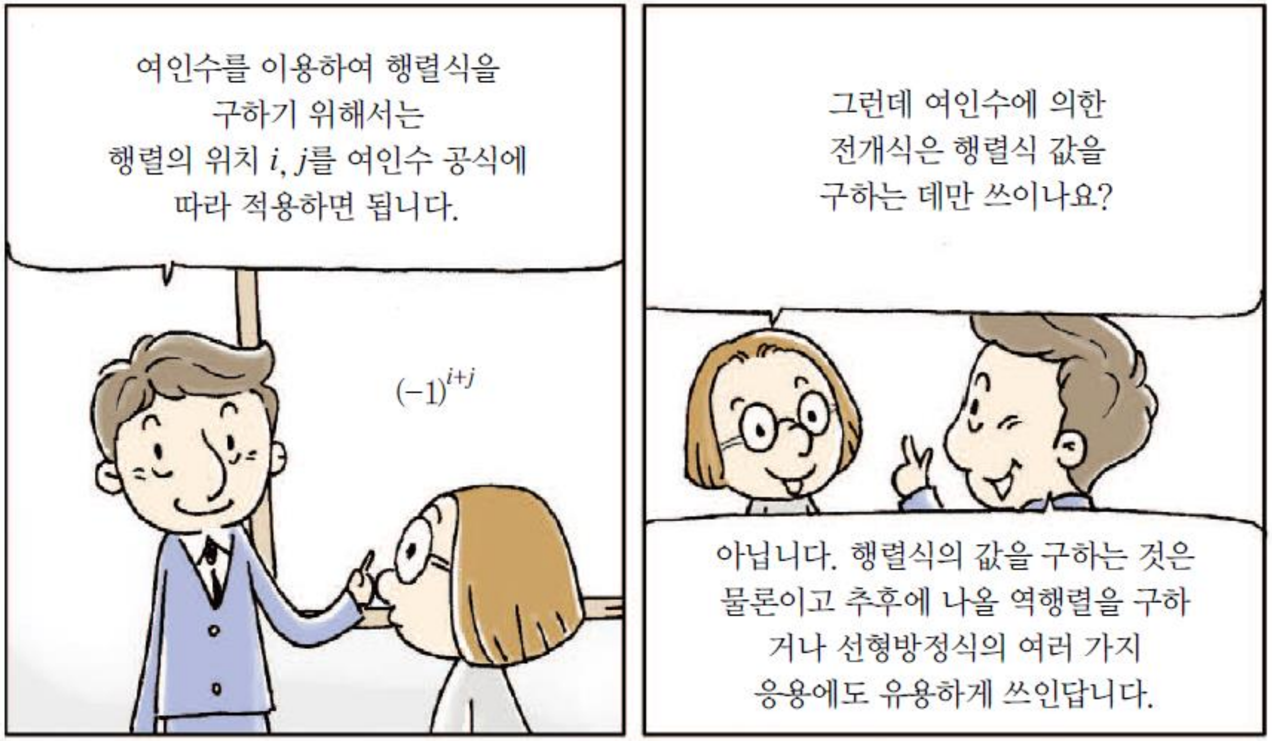
$$A_{12} = (-1)^{1+2} \begin{vmatrix} 6 & 3 \\ 1 & 3 \end{vmatrix}$$

$$A_{13} = (-1)^{1+3} \begin{vmatrix} 6 & 0 \\ 1 & 5 \end{vmatrix}$$

따라서 구하고자 하는 행렬식 Det(A)는 다음과 같다.

$$\begin{aligned} \mathrm{Det}(A) &= 2(-1)^{1+1} \begin{vmatrix} 0 & 3 \\ 5 & 3 \end{vmatrix} + 4(-1)^{1+2} \begin{vmatrix} 6 & 3 \\ 1 & 3 \end{vmatrix} + 7(-1)^{1+3} \begin{vmatrix} 6 & 0 \\ 1 & 5 \end{vmatrix} \\ &= 2(0 \cdot 3 - 3 \cdot 5) - 4(6 \cdot 3 - 3 \cdot 1) + 7(6 \cdot 5 - 0 \cdot 1) \\ &= 120 \quad \blacksquare \end{aligned}$$

3.1 행렬식의 개념과 여인수



3.1 행렬식의 개념과 여인수

예제 ❸-4 다음의 주어진 행렬 A에 대한 행렬식의 값을 구해 보자.

$$A=\begin{bmatrix} 2 & 1 & 0 \\ 1 & 1 & 4 \\ -3 & 2 & 6 \end{bmatrix}$$

풀이 제1행에 대해 M_{ij}를 구하면

$M_{11}=\begin{vmatrix} 1 & 4 \\ 2 & 6 \end{vmatrix}$, $M_{12}=\begin{vmatrix} 1 & 4 \\ -3 & 6 \end{vmatrix}$, $M_{13}=\begin{vmatrix} 1 & 1 \\ -3 & 2 \end{vmatrix}$이다. 따라서 A의 행렬식의 값은

$$\begin{aligned} \mathrm{Det}(A) &= 2(-1)^{1+1}\begin{vmatrix} 1 & 4 \\ 2 & 6 \end{vmatrix} + 1(-1)^{1+2}\begin{vmatrix} 1 & 4 \\ -3 & 6 \end{vmatrix} + 0(-1)^{1+2}\begin{vmatrix} 1 & 1 \\ -3 & 2 \end{vmatrix} \\ &= 2(6-8) - 1(6+12) + 0 \\ &= -22 \end{aligned}$$

가 된다. ■

예제 ❸-5 다음의 주어진 행렬 A에 대한 행렬식의 값을 구해 보자.

$$A=\begin{bmatrix} 1 & 2 & -1 \\ -2 & 0 & 7 \\ 3 & 0 & 7 \end{bmatrix}$$

풀이

$$\operatorname{Det}\begin{bmatrix} 1 & 2 & -1 \\ -2 & 0 & 7 \\ 3 & 0 & 7 \end{bmatrix}=1(-1)^{1+1}\begin{vmatrix} 0 & 7 \\ 0 & 7 \end{vmatrix}+2(-1)^{1+2}\begin{vmatrix} -2 & 7 \\ 3 & 7 \end{vmatrix}$$

$$+(-1)(-1)^{1+3}\begin{vmatrix} -2 & 0 \\ 3 & 0 \end{vmatrix}$$

$$=1(0-0)-2(-14-21)-(0-0)$$

$$=70 \quad ■$$

3.1 행렬식의 개념과 여인수

여기서 잠깐!!

행렬식을 구할 때는 어떤 행을 기준으로 여인수들로 전개해도 좋고, 어떤 열을 기준으로 여인수들로 전개해도 좋은데 그 결과는 같다. 다만 중간에 0이 많이 들어 있는 행이나 열을 기준으로 하면 계산량이 줄어들므로 훨씬 간편하다.

예제 ❸-6 다음 행렬식을 행이 아닌 열에 대해 여인수들로 전개하여 구해 보자.

$$\operatorname{Det}(A)=\begin{vmatrix} 3 & 0 & 1 \\ 1 & 2 & 5 \\ -1 & 4 & 2 \end{vmatrix}$$

풀이 제2열에 대해 전개하여 계산하면 다음과 같다.

$$\begin{vmatrix} 3 & 0 & 1 \\ 1 & 2 & 5 \\ -1 & 4 & 2 \end{vmatrix} \quad \begin{vmatrix} 3 & 0 & 1 \\ 1 & 2 & 5 \\ -1 & 4 & 2 \end{vmatrix} \quad \begin{vmatrix} 3 & 0 & 1 \\ 1 & 2 & 5 \\ -1 & 4 & 2 \end{vmatrix}$$

이 행렬식은

$$(-1)^{1+2}0\begin{vmatrix} 1 & 5 \\ -1 & 2 \end{vmatrix}+(-1)^{2+2}2\begin{vmatrix} 3 & 1 \\ -1 & 2 \end{vmatrix}+(-1)^{3+2}4\begin{vmatrix} 3 & 1 \\ 1 & 5 \end{vmatrix}$$

$$=2\{6-(-1)\}-4(15-1)$$

$$=-42$$

와 같다. 여기서는 특히 제2열에 0이 포함되어 있으므로 전개 과정에서 한 항이 0이 되어 계산 과정이 줄어든다.

또한 위 행렬식을 제3열에 대해 전개하여 계산할 수도 있다.

$$\begin{vmatrix} 3 & 0 & 1 \\ 1 & 2 & 5 \\ -1 & 4 & 2 \end{vmatrix} \qquad \begin{vmatrix} 3 & 0 & 1 \\ 1 & 2 & 5 \\ -1 & 4 & 2 \end{vmatrix} \qquad \begin{vmatrix} 3 & 0 & 1 \\ 1 & 2 & 5 \\ -1 & 4 & 2 \end{vmatrix}$$

이 경우의 행렬식은 다음과 같다.

$$(-1)^{1+3}1\begin{vmatrix} 1 & 2 \\ -1 & 4 \end{vmatrix} + (-1)^{2+3}5\begin{vmatrix} 3 & 0 \\ -1 & 4 \end{vmatrix} + (-1)^{3+3}2\begin{vmatrix} 3 & 0 \\ 1 & 2 \end{vmatrix} = -42$$

따라서 행렬식의 값을 구할 때 어느 열을 선택하여 여인수들로 전개를 하여도 그 결과 값이 같다는 것을 확인할 수 있다. ■

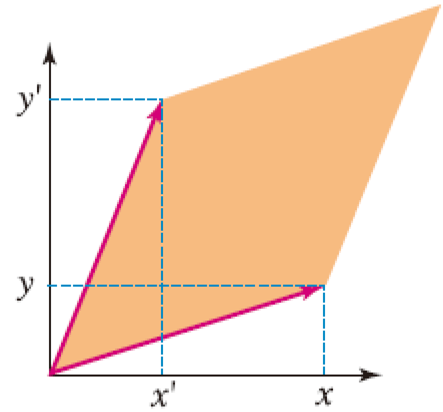
3.1 행렬식의 개념과 여인수

정의 ❸-3 | $n \times n$ 정방행렬 A의 행렬식 $|A|$의 값이 0이 아닐 때 A를 **정칙행렬(non-singular matrix)**이라고 하고, $|A| = 0$일 때 A를 **특이행렬(singular matrix)**이라고 한다.

여기서 잠깐!!

$n \times n$ 행렬 A, B가 정칙행렬인 경우를 가역적(nonsingular, invertible)이라고도 하는데, $AB = BA = I$인 경우를 말한다.

$$\begin{vmatrix} 1 & 4 & 2 & 3 \\ 0 & 1 & 4 & 4 \\ -1 & 0 & 1 & 0 \\ 2 & 0 & 4 & 1 \end{vmatrix} = -\begin{vmatrix} -1 & 0 & 1 & 0 \\ 0 & 1 & 4 & 4 \\ 1 & 4 & 2 & 3 \\ 2 & 0 & 4 & 1 \end{vmatrix}$$



3.2 행렬식의 일반적인 성질

정리 ❸−1 $n \times n$ 행렬 A에서 임의의 두 행(또는 열)이 같으면 행렬식의 값은 0이다. 예를 들어, 다음과 같이 두 행이 같은 행렬식을 살펴보자.

$$A = \begin{vmatrix} 4 & 5 & -1 \\ 3 & 3 & 3 \\ 3 & 3 & 3 \end{vmatrix}$$

이 행렬식은 두 번째 행과 세 번째 행이 같기 때문에 행렬식

$$\mathrm{Det}\,(A) = \begin{vmatrix} 4 & 5 & -1 \\ 3 & 3 & 3 \\ 3 & 3 & 3 \end{vmatrix} = 0 \text{이 된다.}$$

이것은 여인수 전개를 이용하여 계산하면 쉽게 확인할 수 있다.

$$\mathrm{Det}\,(A) = 4\begin{vmatrix} 3 & 3 \\ 3 & 3 \end{vmatrix} - 5\begin{vmatrix} 3 & 3 \\ 3 & 3 \end{vmatrix} - 1\begin{vmatrix} 3 & 3 \\ 3 & 3 \end{vmatrix} = 0$$

3.2 행렬식의 일반적인 성질

정리 ❸-2 $n \times n$ 행렬 A의 임의의 두 행(열)을 서로 바꾸어서 만들어진 행렬을 B라고 하면 $\mathrm{Det}(B) = -\mathrm{Det}(A)$이다.

예를 들어, 1열과 2열을 서로 바꾼 행렬식은 값은 같고 부호만 바뀐다.

$$\begin{vmatrix} 1 & 2 \\ 3 & 4 \end{vmatrix} = -\begin{vmatrix} 2 & 1 \\ 4 & 3 \end{vmatrix} = -2$$

또한 행렬 B가 행렬 $A = \begin{bmatrix} 2 & 1 & 3 \\ 5 & 4 & 6 \\ 8 & 7 & 9 \end{bmatrix}$의 첫째 열과 둘째 열을 바꾸어서 만들어진 행렬이라면 $\mathrm{Det}(B) = -\mathrm{Det}(A)$이다.

$$\mathrm{Det}(B) = \begin{vmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{vmatrix} = -\begin{vmatrix} 2 & 1 & 3 \\ 5 & 4 & 6 \\ 8 & 7 & 9 \end{vmatrix} = -\mathrm{Det}(A)$$

3.2 행렬식의 일반적인 성질

정리 ❸-3 행렬 A의 행렬식의 값은 그 전치행렬의 행렬식의 값과 같다.
즉, $\mathrm{Det}(A) = \mathrm{Det}(A^T)$이다.

예를 들어, $\mathrm{Det}\begin{bmatrix} a & b \\ c & d \end{bmatrix} = \mathrm{Det}\begin{bmatrix} a & c \\ b & d \end{bmatrix}$이고, $\begin{vmatrix} 1 & 2 \\ 3 & 4 \end{vmatrix} = \begin{vmatrix} 1 & 3 \\ 2 & 4 \end{vmatrix} = -2$이다.

3.2 행렬식의 일반적인 성질

정리 ❸-4 A와 B가 $n \times n$ 행렬이면 곱의 행렬식은 행렬식의 곱과 같다.

즉, $\mathrm{Det}(AB) = \mathrm{Det}(A) \cdot \mathrm{Det}(B)$가 성립한다.

예를 들어, 행렬 $A = \begin{bmatrix} 1 & -1 & 2 \\ 3 & 1 & 4 \\ 0 & -2 & 5 \end{bmatrix}$와 $B = \begin{bmatrix} 1 & -2 & 3 \\ 0 & -1 & 4 \\ 2 & 0 & -2 \end{bmatrix}$에 대해

각각의 행렬식을 구하면 $\mathrm{Det}(A) = 16$, $\mathrm{Det}(B) = -8$이다.

그리고 $AB = \begin{bmatrix} 5 & -1 & -5 \\ 11 & -7 & 5 \\ 10 & 2 & -18 \end{bmatrix}$이므로

$\mathrm{Det}(AB) = -128$이고,

$\mathrm{Det}(A) \cdot \mathrm{Det}(B) = 16 \cdot (-8) = -128$이다.

따라서 $\mathrm{Det}(AB) = \mathrm{Det}(A) \cdot \mathrm{Det}(B)$가 성립한다.

3.2 행렬식의 일반적인 성질

정리 ❸-5 행렬식의 어떤 행(또는 열)의 각 원소에 같은 수 k를 곱하여 얻은 행렬식은 처음 행렬식에 k를 곱한 것과 같다. 예를 들어, 행렬식 $\begin{vmatrix} 2 & 3 \\ -1 & 1 \end{vmatrix}$의 1열에 5를 곱하거나 행렬식 $\begin{vmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{vmatrix}$의 2행에 k를 곱하면 다음과 같은 식이 성립한다.

$$\begin{vmatrix} 5\times 2 & 3 \\ 5\times(-1) & 1 \end{vmatrix} = 5\begin{vmatrix} 2 & 3 \\ -1 & 1 \end{vmatrix} = 25$$

$$\begin{vmatrix} a_{11} & a_{12} \\ ka_{21} & ka_{22} \end{vmatrix} = k\begin{vmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{vmatrix}$$

정리 ❸−6 $n \times n$ 행렬 A의 한 행(열)에 있는 모든 원소가 0이면 $\mathrm{Det}(A)=0$이다.

예를 들어, $A=\begin{vmatrix} 0 & 0 \\ 5 & -2 \end{vmatrix}$인 경우 제1행의 항들이 모두 0이므로 $\begin{vmatrix} 0 & 0 \\ 5 & -2 \end{vmatrix}=0$이고,

$A=\begin{vmatrix} 1 & 2 & 0 \\ 1 & 3 & 0 \\ 7 & -1 & 0 \end{vmatrix}$인 경우 제3열의 항들이 모두 0이므로

$\begin{vmatrix} 1 & 2 & 0 \\ 1 & 3 & 0 \\ 7 & -1 & 0 \end{vmatrix}=0$이다.

3.2 행렬식의 일반적인 성질

정리 ❸-7 A를 $n \times n$ 삼각행렬(상부삼각행렬 또는 하부삼각행렬)이라고 하면, Det(A)는 주대각선상의 원소들인 $a_{11}, a_{22}, \cdots, a_{nn}$을 모두 곱한 값과 일치한다. 즉,

$$\mathrm{Det}(A) = a_{11} \times a_{22} \times \cdots \times a_{nn}$$

이다.

증명 n에 대한 수학적 귀납법으로 증명할 수 있으나, 여기서는 $n=3$인 경우에 대해 입증한다.

3×3인 하부삼각행렬 A에서 Det(A)를 제1행에 따라 여인수들로 전개하면 다음과 같다.

$$A = \begin{bmatrix} a_{11} & 0 & 0 \\ a_{21} & a_{22} & 0 \\ a_{31} & a_{32} & a_{33} \end{bmatrix}$$

$$\mathrm{Det}(A) = a_{11} \begin{vmatrix} a_{22} & 0 \\ a_{32} & a_{33} \end{vmatrix} = a_{11}(a_{22}a_{33} - 0 \cdot a_{32}) = a_{11}a_{22}a_{33}$$

따라서 위의 정리가 성립한다. ■

3.2 행렬식의 일반적인 성질

예제 3-7 대각행렬과 상부(하부)삼각행렬의 행렬식은 암산으로도 가능한지를 살펴보자.

풀이 (1) 대각행렬 $A = \begin{bmatrix} 2 & 0 & 0 \\ 0 & 3 & 0 \\ 0 & 0 & 4 \end{bmatrix}$의 행렬식은

$$\mathrm{Det}\,(A) = \begin{vmatrix} 2 & 0 & 0 \\ 0 & 3 & 0 \\ 0 & 0 & 4 \end{vmatrix} = 2 \times 3 \times 4 = 24\text{이다.}$$

(2) 하부삼각행렬 $A = \begin{bmatrix} 2 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 2 & 9 & -3 & 0 \\ 3 & 4 & 5 & 2 \end{bmatrix}$의 행렬식은

$$\mathrm{Det}(A) = \begin{vmatrix} 2 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 2 & 9 & -3 & 0 \\ 3 & 4 & 5 & 2 \end{vmatrix} = 2 \times 1 \times (-3) \times 2 = -12$$이다.

(3) I가 단위행렬이면 $\mathrm{Det}(I) = \begin{vmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{vmatrix} = 1$이다. ■

정리 ❸-8 상부삼각행렬과 하부삼각행렬이 가역적(invertible)이기 위한 필요충분조건은 그 대각선상의 모든 성분들이 0이 아니어야 한다.

예를 들어, 다음과 같이 주대각선상에 0이 있는 상부삼각행렬의 행렬식의 값은 무조건 0이 된다. 따라서 이 경우에는 $|A| = 0$이므로 가역적일 수가 없다.

$$\begin{vmatrix} 7 & 1 & 5 \\ 0 & 0 & -3 \\ 0 & 0 & -2 \end{vmatrix} = 0$$

그러므로 상부삼각행렬과 하부삼각행렬이 가역적(invertible)이기 위한 필요충분조건은 그 대각선상의 모든 성분들이 0이 아니어야 한다.

3.2 행렬식의 일반적인 성질

정리 ❸-9 기본 행 연산에서의 행렬식의 성질은 다음과 같다.

(1) 한 개의 행(또는 열)에 k배를 한 행렬식은 원래 행렬식의 k배와 같다.

(2) 두 개의 행(또는 열)을 교환한 행렬식은 원래 행렬식에서 부호만 바뀐다.

(3) 한 개의 행(또는 열)에 k배를 하여 다른 행(또는 열)에 더하여 만든 행렬식은 원래의 행렬식과 같다.

예제 ❸-8

행렬식의 성질을 이용하여 다음 행렬식의 값을 구해 보자.

$$\mathrm{Det}(A) = \begin{vmatrix} 1 & 2 & 3 \\ 2 & 3 & 5 \\ 3 & -3 & 7 \end{vmatrix}$$

풀이 다음과 같이 여러 번의 기본 행 연산을 통하여 행 사다리꼴로 만든 후 주대각선상에 있는 원소들을 곱하기만 하면 행렬식의 값을 비교적 쉽게 구할 수 있다.

$$\mathrm{Det}(A) = \begin{vmatrix} 1 & 2 & 3 \\ \boxed{2} & 3 & 5 \\ 3 & -3 & 7 \end{vmatrix}$$

$(-2)\times R_1 + R_2 \rightarrow R_2$

(1행에다 -2를 곱하여 2행에 더한다.)

3.2 행렬식의 일반적인 성질

$$= \begin{vmatrix} 1 & 2 & 3 \\ 0 & -1 & -1 \\ \boxed{3} & -3 & 7 \end{vmatrix}$$

$(-3)\times R_1 + R_3 \rightarrow R_3$

(1행에다 −3을 곱하여 3행에 더한다.)

$$= \begin{vmatrix} 1 & 2 & 3 \\ 0 & -1 & -1 \\ 0 & \boxed{-9} & -2 \end{vmatrix}$$

$(-9)\times R_2 + R_3 \rightarrow R_3$

(2행에다 −9를 곱하여 3행에 더한다.)

$$= \begin{vmatrix} 1 & 2 & 3 \\ 0 & -1 & -1 \\ 0 & 0 & 7 \end{vmatrix}$$

$$= 1 \times (-1) \times 7 = -7 \quad \blacksquare$$

여기서 잠깐 !! 제2장에서 언급한 바와 같이 $R_1 \leftrightarrow R_2$, $(-2)\times R_3 \rightarrow R_3$, $(-3)\times R_2 + R_3 \rightarrow R_3$ 등은 일반적으로 통용되는 간결한 표현인데, 이 책에서도 이와 같은 표기법을 병행하여 사용한다.

3.2 행렬식의 일반적인 성질

C program, MATLAB

예제 3-9 행렬식의 성질을 이용하여 다음 행렬식의 값을 구해 보자.

$$|A| = \begin{vmatrix} 2 & 1 & 3 & 4 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \\ 4 & 2 & 1 & -1 \end{vmatrix}$$

풀이 기본 행 연산을 통하여 행 사다리꼴로 변형시켜 삼각행렬로 만들면 그 값을 쉽게 구할 수 있다.

$$|A| = \begin{vmatrix} 2 & 1 & 3 & 4 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \\ 4 & 2 & 1 & -1 \end{vmatrix}$$

$R_1 \leftrightarrow R_3$

(행 교환을 하면 부호가 바뀐다.)

$$=-\begin{vmatrix} 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 \\ \boxed{2} & 1 & 3 & 4 \\ 4 & 2 & 1 & -1 \end{vmatrix} \qquad (-2)\times R_1 + R_3 \rightarrow R_3$$

$$=-\begin{vmatrix} 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 2 \\ \boxed{4} & 2 & 1 & -1 \end{vmatrix} \qquad (-4)\times R_1 + R_4 \rightarrow R_4$$

$$=-\begin{vmatrix} 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & \boxed{1} & 1 & 2 \\ 0 & 2 & -3 & -5 \end{vmatrix} \qquad (-1)\times R_2 + R_3 \rightarrow R_3$$

$$=-\begin{vmatrix} 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 2 \\ 0 & \boxed{2} & -3 & 5 \end{vmatrix}$$

$(-2)\times R_2+R_4\rightarrow R_4$

$$=-\begin{vmatrix} 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 2 \\ 0 & 0 & -5 & 5 \end{vmatrix}$$

$R_3\leftrightarrow R_4$

(행 교환을 하면 부호가 바뀐다.)

$$=+\begin{vmatrix} 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & -5 & 5 \\ 0 & 0 & 0 & 2 \end{vmatrix}$$

삼각행렬이 되었으므로 주대각선상의 원소들을 모두 곱한다.

$$=1\times1\times(-5)\times2=-10 \quad ■$$

역행렬

$$M^{-1} = \frac{1}{ad - bc}\begin{bmatrix} d & -b \\ -c & a \end{bmatrix}$$

$$\frac{1}{\text{행렬식}}$$

$$\begin{bmatrix} d & -b \\ -c & a \end{bmatrix}$$

부호만 바뀐다

자리가 바뀐다

$$[A \mid I] = \left[\begin{array}{ccc|ccc} 1 & 2 & 3 & 1 & 0 & 0 \\ 2 & 5 & 3 & 0 & 1 & 0 \\ 1 & 0 & 8 & 0 & 0 & 1 \end{array}\right] \longrightarrow \left[\begin{array}{ccc|ccc} 1 & 0 & 0 & -40 & 16 & 9 \\ 0 & 1 & 0 & 13 & -5 & -3 \\ 0 & 0 & 1 & 5 & -2 & -1 \end{array}\right]$$

역행렬

정의 ❸-4 행렬 A와 B가 모두 $n \times n$ 행렬일 때, $AB = BA = I$(I : 항등행렬)인 행렬 B가 존재하는 경우 A를 **가역적(nonsingular, invertible)**이라 한다. 이 경우 $AB = BA = I$가 성립하는 하나뿐인 행렬 B를 A의 **역행렬(inverse matrix)**이라고 하고 A^{-1}로 나타내는데, $AA^{-1} = A^{-1}A = I$가 항상 성립한다.

가역적인 행렬, 즉 역행렬이 존재하는 행렬을 **정칙행렬(nonsingular matrix)**이라고 하고, 그렇지 않은 행렬을 **특이행렬(singular matrix)**이라고 한다.

3.3 역행렬

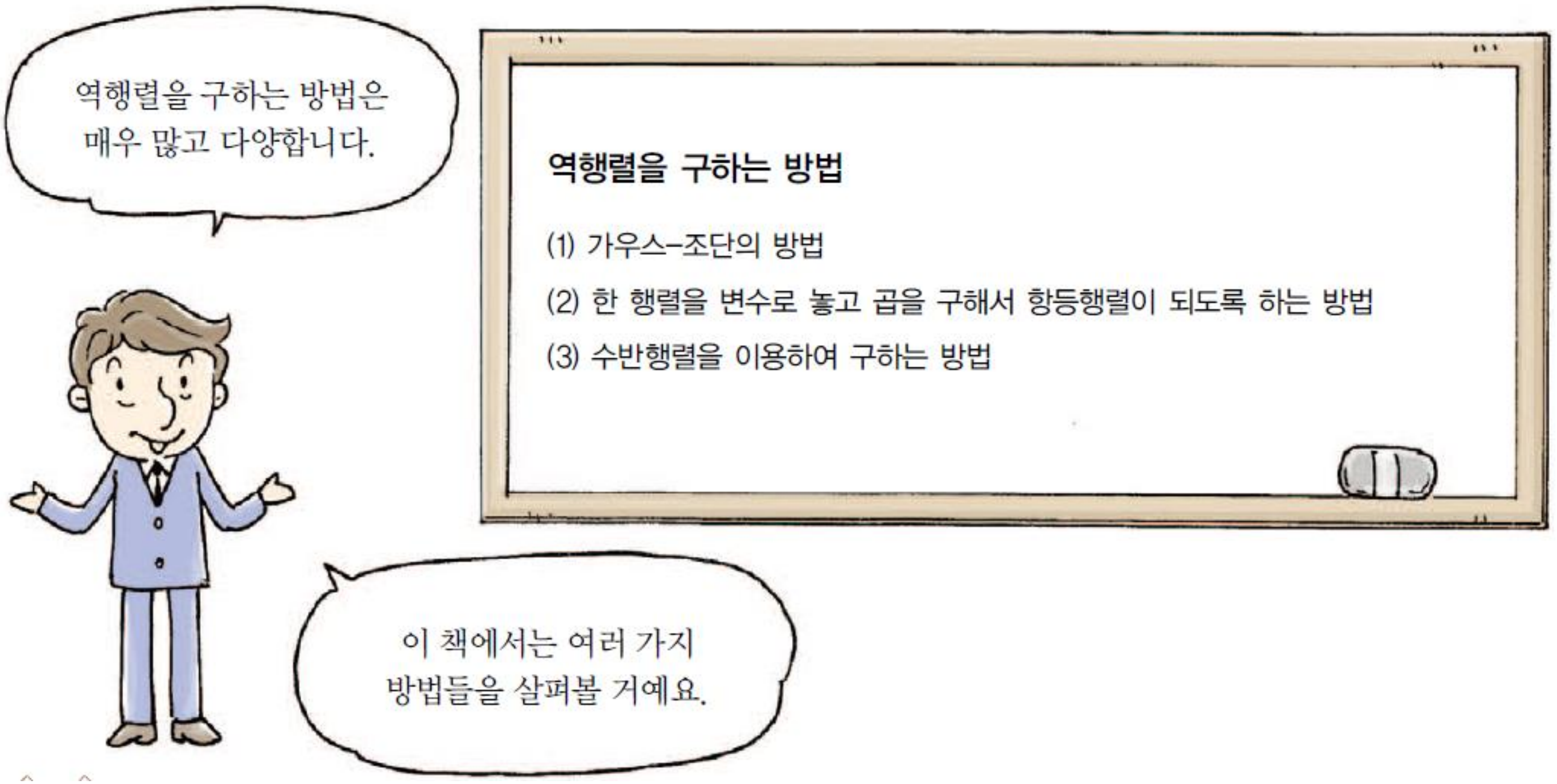
정리 ❸-10 행렬 A에서 $AB=BA=I$인 행렬 B가 존재하면 그것은 오직 하나뿐이다.

증명 $AC=CA=I$인 행렬 C가 존재한다면

$$C=CI=C(AB)=(CA)B=IB=B$$

따라서 $C=B$가 되므로 B는 유일하다. ■

3.3 역행렬


역행렬을 구하는 방법은 매우 많고 다양합니다.
역행렬을 구하는 방법
(1) 가우스–조단의 방법
(2) 한 행렬을 변수로 놓고 곱을 구해서 항등행렬이 되도록 하는 방법
(3) 수반행렬을 이용하여 구하는 방법
이 책에서는 여러 가지 방법들을 살펴볼 거예요.

예제 ❸-10 다음의 행렬 A와 B의 곱이 항등행렬이 됨을 보임으로써 가역적임을 입증해 보자.

$$A=\begin{bmatrix}2 & 5\\1 & 3\end{bmatrix},\quad B=\begin{bmatrix}3 & -5\\-1 & 2\end{bmatrix}$$

풀이

$$AB=\begin{bmatrix}6-5 & -10+10\\3-3 & -5+6\end{bmatrix}=\begin{bmatrix}1 & 0\\0 & 1\end{bmatrix}$$

$$BA=\begin{bmatrix}6-5 & 15-15\\-2+2 & -5+6\end{bmatrix}=\begin{bmatrix}1 & 0\\0 & 1\end{bmatrix}$$

AB와 BA의 결과가 항등행렬이 되므로 행렬 A와 B는 가역적이다. ■

예제 3-11 다음의 행렬 A와 B가 서로 역행렬 관계임을 입증해 보자.

$$A=\begin{bmatrix}1 & 0 & 2\\ 2 & -1 & 3\\ 4 & 1 & 8\end{bmatrix},\quad B=\begin{bmatrix}-11 & 2 & 2\\ -4 & 0 & 1\\ 6 & -1 & -1\end{bmatrix}$$

풀이 행렬의 곱 AB를 구하면,

$$AB=\begin{bmatrix}-11+0+12 & 2+0-2 & 2+0-2\\ -22+4+18 & 4+0-3 & 4-1-3\\ -44-4+48 & 8+0-8 & 8+1-8\end{bmatrix}=\begin{bmatrix}1 & 0 & 0\\ 0 & 1 & 0\\ 0 & 0 & 1\end{bmatrix}=I$$

$AB=I$이기 때문에 정의에 따라 A와 B는 서로 역행렬이다. ■

예제 ❸-12

역행렬을 구하는 방법 중 하나는 역행렬을 변수로 놓고 서로 곱하여 항등행렬이 되도록 하는 방법이다. 가령

$$A=\begin{bmatrix}1 & 2\\ 3 & 4\end{bmatrix} \text{이면 } A^{-1}=\begin{bmatrix}a & b\\ c & d\end{bmatrix} \text{라 한다.}$$

$$AA^{-1}=\begin{bmatrix}1 & 2\\ 3 & 4\end{bmatrix}\begin{bmatrix}a & b\\ c & d\end{bmatrix}=\begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix}$$

따라서 $\begin{bmatrix}a+2c & b+2d\\ 3a+4c & 3b+4d\end{bmatrix}=\begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix}$

$$\begin{aligned}
a \qquad\quad + 2c \qquad\quad &= 1\\
b \qquad\quad + 2d &= 0\\
3a \qquad\quad + 4c \qquad\quad &= 0\\
3b \qquad\quad + 4d &= 1
\end{aligned}$$

이 식을 풀면 $a=-2$, $b=1$, $c=\frac{3}{2}$, $d=-\frac{1}{2}$이다.

따라서 $A^{-1}=\begin{bmatrix} a & b \\ c & d \end{bmatrix}=\begin{bmatrix} -2 & 1 \\ \frac{3}{2} & -\frac{1}{2} \end{bmatrix}$ ■

3.3 역행렬

정리 ❸-11 $n \times n$ 정칙행렬 A, B에 대해 다음과 같은 역행렬의 성질들이 성립한다.

(1) $(A^{-1})^{-1} = A$ (2) $(AB)^{-1} = B^{-1}A^{-1}$

증명 (1) A가 정칙행렬이면 $AA^{-1} = A^{-1}A = I$이므로 A의 역행렬은 A^{-1}이고 A^{-1}의 역행렬 $(A^{-1})^{-1}$은 A이다. 따라서 $(A^{-1})^{-1} = A$이다.

(2) A, B가 정칙행렬이면 역행렬 A^{-1}, B^{-1}가 존재하여 $AA^{-1} = A^{-1}A = I$이고 $BB^{-1} = B^{-1}B = I$이다.

$$(AB)(B^{-1}A^{-1}) = A(BB^{-1})A^{-1} = AIA^{-1} = AA^{-1} = I$$
$$(B^{-1}A^{-1})(AB) = B^{-1}(A^{-1}A)B = B^{-1}IB = B^{-1}B = I$$

그러므로 $(AB)(B^{-1}A^{-1}) = (B^{-1}A^{-1})(AB) = I$

따라서 $B^{-1}A^{-1}$은 AB의 역행렬이다. ■

예제 ❸-13 두 행렬 $A=\begin{bmatrix}1 & 2\\1 & 3\end{bmatrix}$, $B=\begin{bmatrix}3 & 2\\2 & 2\end{bmatrix}$에 대하여 $(AB)^{-1}=B^{-1}A^{-1}$임을 확인해 보자.

풀이 두 개의 식을 계산하여 그 결과가 같음을 보인다.

$A^{-1}=\begin{bmatrix}3 & -2\\-1 & 1\end{bmatrix}$, $B^{-1}=\begin{bmatrix}1 & -1\\-1 & \frac{3}{2}\end{bmatrix}$이고 $AB=\begin{bmatrix}7 & 6\\9 & 8\end{bmatrix}$이므로

$$(AB)^{-1}=\begin{bmatrix}4 & -3\\-\frac{9}{2} & \frac{7}{2}\end{bmatrix}$$

$$B^{-1}A^{-1}=\begin{bmatrix}1 & -1\\-1 & \frac{3}{2}\end{bmatrix}\begin{bmatrix}3 & -2\\-1 & 1\end{bmatrix}=\begin{bmatrix}4 & -3\\-\frac{9}{2} & \frac{7}{2}\end{bmatrix}$$

따라서 $(AB)^{-1}=B^{-1}A^{-1}$이다. ■

여기서 잠깐!! A, B가 정방행렬이면서 가역적일 때 $(A^{-1})^{-1}=A$는 역의 역이라서 원래의 행렬이 되는 것은 대충 짐작이 된다. 하지만 $(AB)^{-1}=B^{-1}A^{-1}$의 의미는 무엇일까? 그것은 두 행렬을 곱한 후에 역행렬을 만들면 원래의 순서와는 반대인 역행렬의 곱으로 나온다는 의미이다.

Find the inverse matrix

$$M^T = \begin{pmatrix} 1 & 0 & 5 \\ 2 & 1 & 6 \\ 3 & 4 & 0 \end{pmatrix}$$

$M_{11} = \begin{vmatrix} 1 & 6 \\ 4 & 0 \end{vmatrix} = -24$ $M_{12} = \begin{vmatrix} 2 & 6 \\ 3 & 0 \end{vmatrix} = -18$ $M_{13} = \begin{vmatrix} 2 & 1 \\ 3 & 4 \end{vmatrix} = 5$

$M_{21} = \begin{vmatrix} 0 & 5 \\ 4 & 0 \end{vmatrix} = -20$ $M_{22} = \begin{vmatrix} 1 & 5 \\ 3 & 0 \end{vmatrix} = -15$ $M_{23} = \begin{vmatrix} 1 & 0 \\ 3 & 4 \end{vmatrix} = 4$

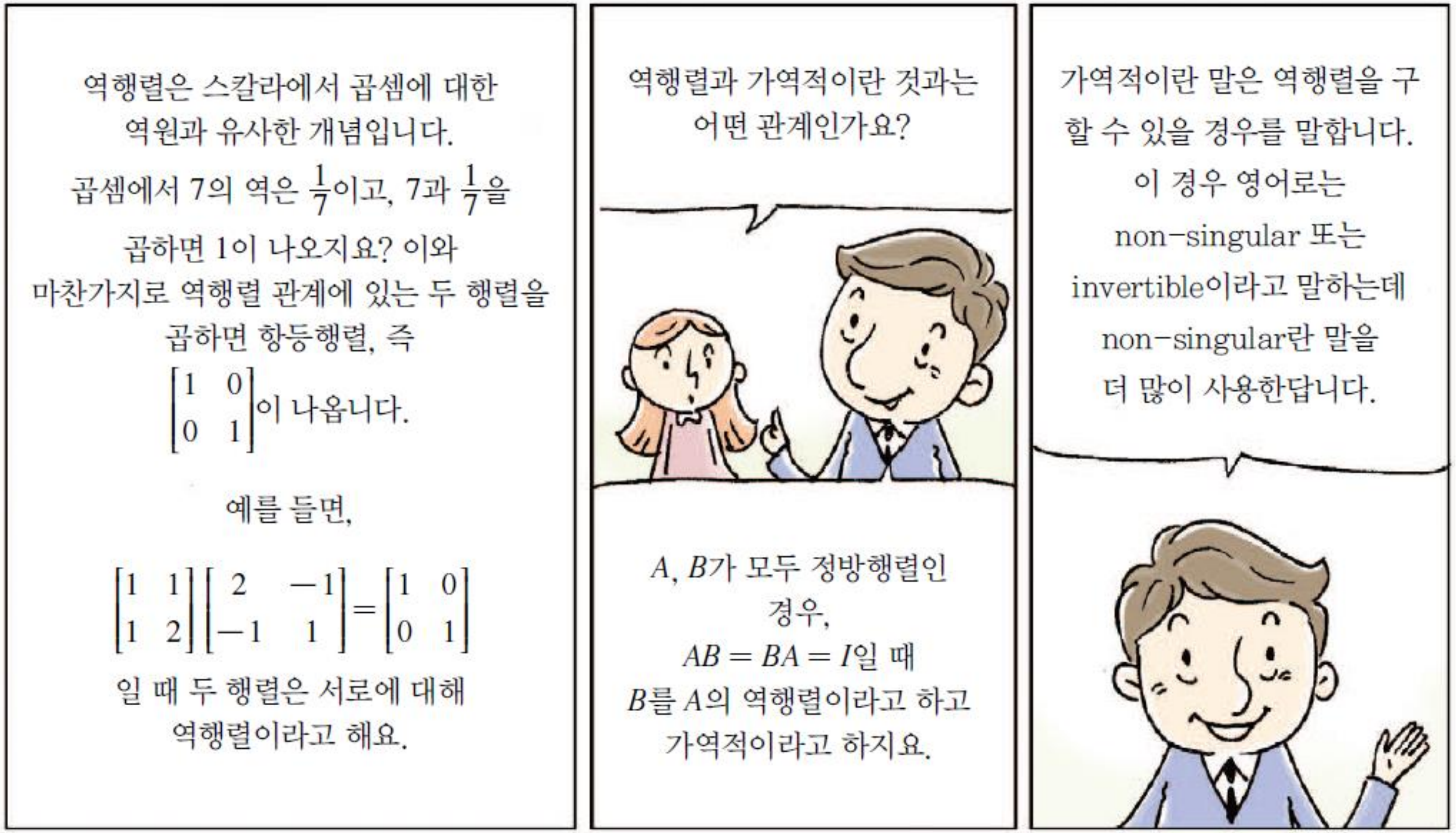
$M_{31} = \begin{vmatrix} 0 & 5 \\ 1 & 6 \end{vmatrix} = -5$ $M_{32} = \begin{vmatrix} 1 & 5 \\ 2 & 6 \end{vmatrix} = -4$ $M_{33} = \begin{vmatrix} 1 & 0 \\ 2 & 1 \end{vmatrix} = 1$

This gives: $\begin{pmatrix} -24 & -18 & 5 \\ -20 & -15 & 4 \\ -5 & -4 & 1 \end{pmatrix}$ Apply $\begin{pmatrix} + & - & + \\ - & + & - \\ + & - & + \end{pmatrix}$ Where + means no sign change and - means a sign change, gives: $\begin{pmatrix} -24 & 18 & 5 \\ 20 & -15 & -4 \\ -5 & 4 & 1 \end{pmatrix}$

Therefore the adj(M) [The adjugate of M] = $\begin{pmatrix} -24 & 18 & 5 \\ 20 & -15 & -4 \\ -5 & 4 & 1 \end{pmatrix}$

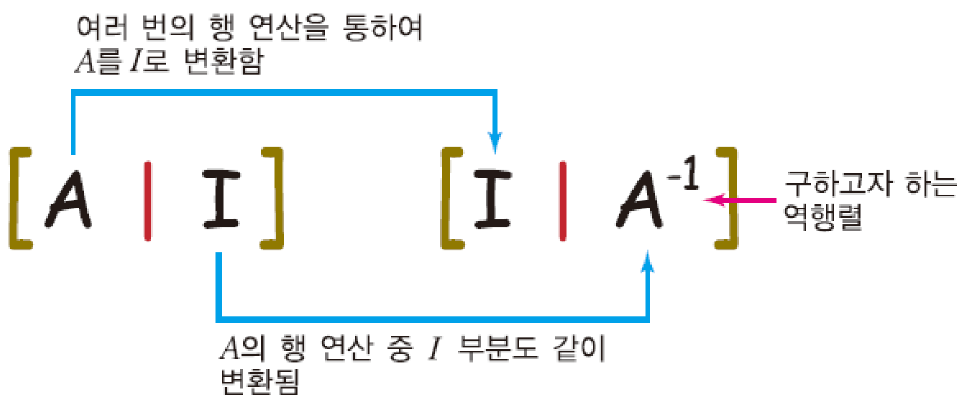
정의 ❸-5 | 주어진 행렬 A의 오른쪽에다 추가적으로 첨가하여(augmented) 만든 행렬을 **첨가 행렬(augmented matrix)**이라고 한다.

3.3 역행렬


역행렬은 스칼라에서 곱셈에 대한 역원과 유사한 개념입니다.
곱셈에서 7의 역은 $\frac{1}{7}$이고, 7과 $\frac{1}{7}$을 곱하면 1이 나오지요? 이와 마찬가지로 역행렬 관계에 있는 두 행렬을 곱하면 항등행렬, 즉 $\begin{bmatrix}1 & 0\\0 & 1\end{bmatrix}$이 나옵니다.
예를 들면,
$\begin{bmatrix}1 & 1\\1 & 2\end{bmatrix}\begin{bmatrix}2 & -1\\-1 & 1\end{bmatrix}=\begin{bmatrix}1 & 0\\0 & 1\end{bmatrix}$
일 때 두 행렬은 서로에 대해 역행렬이라고 해요.
역행렬과 가역적이란 것과는 어떤 관계인가요?
A, B가 모두 정방행렬인 경우, $AB=BA=I$일 때 B를 A의 역행렬이라고 하고 가역적이라고 하지요.
가역적이란 말은 역행렬을 구할 수 있을 경우를 말합니다. 이 경우 영어로는 non-singular 또는 invertible이라고 말하는데 non-singular란 말을 더 많이 사용한답니다.

3.3 역행렬

정리 ❸-12 가우스-조단의 역행렬을 구하는 알고리즘을 이용하여 A의 역행렬인 A^{-1}를 구할 수 있다. 우선 〈그림 3.2〉와 같이 첨가행렬 $[A \mid I]$를 만들고 행 연산을 한다. 만약 행 연산을 하여 $[A \mid I]$를 $[I \mid A^{-1}]$의 형태로 변환했을 경우 A의 역행렬인 A^{-1}를 구할 수 있다.



〈그림 3.2〉 가우스-조단 방식의 역행렬 구하기

3.3 역행렬

가우스-조단의 역행렬을 구하는 알고리즘

단계 1 원래의 A 행렬에다 항등행렬 I를 첨가하여 첨가행렬 $[A \mid I]$로 만든다.

단계 2 행렬 A 부분이 항등행렬로 바뀔 때까지 행 연산을 계속한다.

단계 3 A가 가역적 행렬인지를 결정한다.

(1) A를 항등행렬로 변환할 수 있으면 원래 I 위치에 있는 행렬이 A^{-1}가 된다 $(I \mid A)$.

(2) 만약 A의 행 연산 과정에서 한 행이 모두 0이 되면 A는 비가역적 행렬이므로 역행렬을 구하는 과정을 중단한다.

예제 ❸-14

행렬 $A=\begin{bmatrix}3 & 4\\2 & 3\end{bmatrix}$의 역행렬을 가우스-조단의 방법으로 구해 보자.

풀이 가우스-조단 알고리즘을 적용하여 역행렬을 구한다.

먼저 a_{11}의 값을 1로 만들기 위해 $\frac{1}{3}$을 곱한다.

$$\left[\begin{array}{cc|cc}3 & 4 & 1 & 0\\2 & 3 & 0 & 1\end{array}\right]$$

$\left(\frac{1}{3}\right)\times R_1\rightarrow R_1$

$$\left[\begin{array}{cc|cc}1 & \frac{4}{3} & \frac{1}{3} & 0\\ \boxed{2} & 3 & 0 & 1\end{array}\right]$$

$(-2)\times R_1+R_2\rightarrow R_2$

$$\left[\begin{array}{cc|cc} 1 & \frac{4}{3} & \frac{1}{3} & 0 \\ 0 & \frac{1}{3} & -\frac{2}{3} & 1 \end{array}\right]$$

$3 \times R_2 \rightarrow R_2$

$$\left[\begin{array}{cc|cc} 1 & \frac{4}{3} & \frac{1}{3} & 0 \\ 0 & \boxed{1} & -2 & 3 \end{array}\right]$$

$\left(-\frac{4}{3}\right) \times R_2 + R_1 \rightarrow R_1$

$$\left[\begin{array}{cc|cc} 1 & 0 & 3 & -4 \\ 0 & 1 & -2 & 3 \end{array}\right]$$을 얻는다.

따라서 다음과 같은 역행렬을 구할 수 있다.

$$A^{-1} = \begin{bmatrix} 3 & -4 \\ -2 & 3 \end{bmatrix} \quad \blacksquare$$

3.3 역행렬

예제 ❸-15 기본 행 연산을 이용하여 다음 행렬의 역행렬을 구해 보자.

$$A=\begin{bmatrix}1 & 2 & 3\\ 2 & 5 & 3\\ 1 & 0 & 8\end{bmatrix}$$

풀이 행렬 A의 오른쪽에 항등행렬을 같이 써서 아래와 같은 첨가행렬을 만든다. 이 행렬에 대해 기본 행 연산을 적용하여 왼쪽을 항등행렬로 만들면 오른쪽에 얻어지는 행렬이 바로 A의 역행렬이 된다.

$$\left[\begin{array}{ccc|ccc}1 & 2 & 3 & 1 & 0 & 0\\ \boxed{2} & 5 & 3 & 0 & 1 & 0\\ \boxed{1} & 0 & 8 & 0 & 0 & 1\end{array}\right]$$

$(-2)\times R_1+R_2\rightarrow R_2$
$(-1)\times R_1+R_3\rightarrow R_3$

$$\left[\begin{array}{ccc|ccc}1 & 2 & 3 & 1 & 0 & 0\\ 0 & 1 & -3 & -2 & 1 & 0\\ 0 & \boxed{-2} & 5 & -1 & 0 & 1\end{array}\right]$$

$2\times R_2+R_3\rightarrow R_3$

$$\left[\begin{array}{ccc|ccc} 1 & 2 & 3 & 1 & 0 & 0 \\ 0 & 1 & -3 & -2 & 1 & 0 \\ 0 & 0 & -1 & -5 & 2 & 1 \end{array}\right]$$

$(-1)\times R_3 \rightarrow R_3$

$$\left[\begin{array}{ccc|ccc} 1 & 2 & \boxed{3} & 1 & 0 & 0 \\ 0 & 1 & \boxed{-3} & -2 & 1 & 0 \\ 0 & 0 & 1 & 5 & -2 & -1 \end{array}\right]$$

$(-3)\times R_3 + R_1 \rightarrow R_1$

$3\times R_3 + R_2 \rightarrow R_2$

$$\left[\begin{array}{ccc|ccc} 1 & \boxed{2} & 0 & -14 & 6 & 3 \\ 0 & 1 & 0 & 13 & -5 & -3 \\ 0 & 0 & 1 & 5 & -2 & -1 \end{array}\right]$$

$(-2)\times R_2 + R_1 \rightarrow R_1$

$$\left[\begin{array}{ccc|ccc} 1 & 0 & 0 & -40 & 16 & 9 \\ 0 & 1 & 0 & 13 & -5 & -3 \\ 0 & 0 & 1 & 5 & -2 & -1 \end{array}\right]$$

이 단계에서 왼쪽 행렬이 항등행렬이 되었다. 따라서

$$A^{-1}=\begin{bmatrix}-40 & 16 & 9\\ 13 & -5 & -3\\ 5 & -2 & -1\end{bmatrix}\text{이다.}$$

이 행렬이 실제로 A의 역행렬인지를 확인하기 위해서는 $AA^{-1}=I$를 계산해 보면 된다. ■

예제 ❸-16 $A=\begin{bmatrix}1 & -3 & 4\\ 2 & -5 & 7\\ 0 & -1 & 1\end{bmatrix}$일 때 A^{-1}이 존재할 경우 역행렬을 구해 보자.

풀이 $(A \mid I)$를 이용하여 역행렬을 구하는 방법을 적용하면 다음과 같다.

$$\left[\begin{array}{ccc|ccc}1 & -3 & 4 & 1 & 0 & 0\\ \boxed{2} & -5 & 7 & 0 & 1 & 0\\ 0 & -1 & 1 & 0 & 0 & 1\end{array}\right] \qquad (-2)\times R_1+R_2\rightarrow R_2$$

$$\left[\begin{array}{ccc|ccc}1 & \boxed{-3} & 4 & 1 & 0 & 0\\ 0 & 1 & -1 & -2 & 1 & 0\\ 0 & \boxed{-1} & 1 & 0 & 0 & 1\end{array}\right] \qquad \begin{array}{l}R_2+R_3\rightarrow R_3\\ 3\times R_2+R_1\rightarrow R_1\end{array}$$

$$\left[\begin{array}{ccc|ccc}1 & 0 & 1 & -5 & 3 & 0\\ 0 & 1 & -1 & -2 & 1 & 0\\ 0 & 0 & 0 & -2 & 1 & 1\end{array}\right]$$

왼쪽의 결과로 볼 때 제일 아래 부분의 항들이 모두 0이 되었으므로 더 이상의 행 연산을 진행할 필요가 없다. 즉, 행렬 A 부분의 행 연산 결과가 항등행렬이 될 수 없으므로 A는 가역적일 수가 없고 역행렬을 구할 수 없다. ■

수반행렬에 의한 역행렬 구하기

- 다음과 같이 A의 여인수를 성분으로 가지는 행렬 $B = [Aij]$ 를 행렬 A의 여인수 행렬이라고 할 때 여인수 행렬 B의 전치행렬을 A의 **수반행렬(Adjugate matrix)**이라고 하며 $\boldsymbol{Adj(A)}$와 같이 나타낸다.

$$B = \begin{bmatrix} A_{11} & A_{12} & \cdots & A_{1n} \\ A_{21} & A_{22} & \cdots & A_{2n} \\ \vdots & \vdots & & \vdots \\ A_{m1} & A_{m2} & \cdots & A_{mn} \end{bmatrix}$$

$$\mathrm{Adj}(A) = B^T = \begin{bmatrix} A_{11} & A_{21} & \cdots & A_{n1} \\ A_{12} & A_{22} & \cdots & A_{n2} \\ \vdots & \vdots & & \vdots \\ A_{1m} & A_{2m} & \cdots & A_{mn} \end{bmatrix}$$

예제 ❸-17

$A=\begin{bmatrix} 2 & 4 & 3 \\ 0 & 1 & -1 \\ 3 & 5 & 7 \end{bmatrix}$일 때 Adj$(A)$를 계산해 보자.

풀이 먼저 A의 소행렬식은 다음과 같다.

$$M_{11}=\begin{vmatrix} 1 & -1 \\ 5 & 7 \end{vmatrix}=12 \quad M_{12}=\begin{vmatrix} 0 & -1 \\ 3 & 7 \end{vmatrix}=3 \quad M_{13}=\begin{vmatrix} 0 & 1 \\ 3 & 5 \end{vmatrix}=-3$$

$$M_{21}=\begin{vmatrix} 4 & 3 \\ 5 & 7 \end{vmatrix}=13 \quad M_{22}=\begin{vmatrix} 2 & 3 \\ 3 & 7 \end{vmatrix}=5 \quad M_{23}=\begin{vmatrix} 2 & 4 \\ 3 & 5 \end{vmatrix}=-2$$

$$M_{31}=\begin{vmatrix} 4 & 3 \\ 1 & -1 \end{vmatrix}=-7 \quad M_{32}=\begin{vmatrix} 2 & 3 \\ 0 & -1 \end{vmatrix}=-2 \quad M_{33}=\begin{vmatrix} 2 & 4 \\ 0 & 1 \end{vmatrix}=2$$

따라서 $(-1)^{i+j}$의 값에 따라 부호를 붙인 A의 여인수는 다음과 같다.

$$\begin{array}{lll} A_{11}=\ \ 12 & A_{12}=-3 & A_{13}=-3 \\ A_{21}=-13 & A_{22}=\ \ 5 & A_{23}=\ \ 2 \\ A_{31}=\ -7 & A_{32}=\ \ 2 & A_{33}=\ \ 2 \end{array}$$

그러므로 여인수 행렬 B는

$$B=\begin{bmatrix} 12 & -3 & -3 \\ -13 & 5 & 2 \\ -7 & 2 & 2 \end{bmatrix}\text{이므로}$$

B의 전치행렬을 구하면 된다.

$$\text{Adj}(A)=B^T=\begin{bmatrix} 12 & -13 & -7 \\ -3 & 5 & 2 \\ -3 & 2 & 2 \end{bmatrix}\text{이다. } \blacksquare$$

예제 ❸-18 다음 행렬의 소행렬식, 여인수, 수반행렬을 구해 보자.

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$$

풀이 먼저 A의 소행렬식은 다음과 같이 9개를 구할 수 있다.

$$M_{11} = \begin{vmatrix} 5 & 6 \\ 8 & 9 \end{vmatrix} = 45 - 48 = -3$$

$$M_{12} = \begin{vmatrix} 4 & 6 \\ 7 & 9 \end{vmatrix} = 36 - 42 = -6$$

$$M_{13} = \begin{vmatrix} 4 & 5 \\ 7 & 8 \end{vmatrix} = 32 - 35 = -3$$

$$M_{21}=\begin{vmatrix} 2 & 3 \\ 8 & 9 \end{vmatrix}=18-24=-6$$

$$M_{22}=\begin{vmatrix} 1 & 3 \\ 7 & 9 \end{vmatrix}=9-21=-12$$

$$M_{23}=\begin{vmatrix} 1 & 2 \\ 7 & 8 \end{vmatrix}=8-14=-6$$

$$M_{31}=\begin{vmatrix} 2 & 3 \\ 5 & 6 \end{vmatrix}=12-15=-3$$

$$M_{32}=\begin{vmatrix} 1 & 3 \\ 4 & 6 \end{vmatrix}=6-12=-6$$

$$M_{33}=\begin{vmatrix} 1 & 2 \\ 4 & 5 \end{vmatrix}=5-8=-3$$

따라서 A의 여인수는

$$A_{11}=(-1)^{1+1}(-3)=-3$$
$$A_{21}=(-1)^{2+1}(-6)=6$$
$$A_{31}=(-1)^{3+1}(-3)=-3$$

$$A_{12} = (-1)^{1+2}(-6) = 6$$
$$A_{22} = (-1)^{2+2}(-12) = -12$$
$$A_{32} = (-1)^{3+2}(-6) = 6$$

$$A_{13} = (-1)^{1+3}(-3) = -3$$
$$A_{23} = (-1)^{2+3}(-6) = 6$$
$$A_{33} = (-1)^{3+3}(-3) = -3$$

그러므로 여인수 행렬 $B = \begin{bmatrix} -3 & 6 & -3 \\ 6 & -12 & 6 \\ -3 & 6 & -3 \end{bmatrix}$이 된다.

따라서 수반행렬은 B의 전치행렬이 된다.

$\mathrm{Adj}(A) = B^T = \begin{bmatrix} -3 & 6 & -3 \\ 6 & -12 & 6 \\ -3 & 6 & -3 \end{bmatrix}$이다. ■

수반행렬을 이용하여 역행렬을 구하는 방법

(1) 소행렬식, 여인수를 구한다.

(2) $A_{ij}=(-1)^{i+j}M_{ij}$를 이용한다.

(3) 여인수 행렬의 전치행렬을 구한다.

3.3 역행렬

정리 ❸–13 A가 $n \times n$ 행렬일 때 다음의 2가지가 성립한다.

(1) A가 가역적이기 위한 필요충분조건은 $\mathrm{Det}(A) \neq 0$이다.

(2) $\mathrm{Det}(A) \neq 0$이면 A의 역행렬은 다음과 같다.

$$A^{-1} = \frac{1}{\mathrm{Det}(A)} \mathrm{Adj}(A)$$

2×2 행렬에 대한 역행렬은 이 정리의 특별한 경우로, $\mathrm{Det}(A) \neq 0$이면 다음과 같은 방법으로 A의 역행렬을 구하는 것이 매우 편리하다.

$$A^{-1} = \frac{1}{\mathrm{Det}(A)} \begin{bmatrix} a_{22} & -a_{12} \\ -a_{21} & a_{11} \end{bmatrix}$$

예제 3-19 $A=\begin{bmatrix}1 & 4\\ 2 & 10\end{bmatrix}$의 역행렬을 구해 보자.

풀이 $\text{Det}(A)=10-8=2\neq 0$이므로 A는 가역적이다. 따라서 A^{-1}는 위의 간단한 역행렬 공식에 따라 다음과 같이 구할 수 있다.

$$A^{-1}=\frac{1}{2}\begin{bmatrix}10 & -4\\ -2 & 1\end{bmatrix}=\begin{bmatrix}5 & -2\\ -1 & \frac{1}{2}\end{bmatrix} \blacksquare$$

C program, MATLAB

예제 ❸-20

$A=\begin{bmatrix}3 & 1\\4 & 2\end{bmatrix}$일 때 수반행렬을 이용하여 A의 역행렬을 구해 보자.

풀이 $|A|=\begin{vmatrix}3 & 1\\4 & 2\end{vmatrix}=2$이므로 가역적이다. 그리고 여인수를 구하면 다음과 같다.

$$A_{11}=(-1)^{1+1}\cdot 2=2 \qquad A_{12}=(-1)^{1+2}\cdot 4=-4$$
$$A_{21}=(-1)^{2+1}\cdot 1=-1 \qquad A_{22}=(-1)^{2+2}\cdot 3=3$$

그러므로 A의 여인수 행렬 B는

$B=\begin{bmatrix}2 & -4\\-1 & 3\end{bmatrix}$이므로 $\mathrm{Adj}(A)=\begin{bmatrix}2 & -1\\-4 & 3\end{bmatrix}$이다.

따라서 $A^{-1}=\frac{1}{|A|}\begin{bmatrix}2 & -1\\-4 & 3\end{bmatrix}=\frac{1}{2}\begin{bmatrix}2 & -1\\-4 & 3\end{bmatrix}$이 된다. ■

C program, MATLAB

예제 ③-21

다음에서 행렬식과 Adj(A)를 각각 구하고 역행렬이 존재하면 역행렬을 구해 보자.

$$(1)\ A=\begin{bmatrix}2&-3&1\\5&4&1\\2&-2&-1\end{bmatrix}\quad (2)\ A=\begin{bmatrix}3&1&0\\2&-1&1\\5&5&-7\end{bmatrix}\quad (3)\ A=\begin{bmatrix}2&-1&4\\5&-2&9\\3&2&-1\end{bmatrix}$$

풀이 먼저 행렬식을 구하고 Adj(A)를 만든 후 역행렬을 구한다.

$$(1)\ |A|=2(-1)^{1+1}\begin{vmatrix}4&1\\-2&-1\end{vmatrix}-3(-1)^{1+2}\begin{vmatrix}5&1\\2&-1\end{vmatrix}+1(-1)^{1+3}\begin{vmatrix}5&4\\2&-2\end{vmatrix}$$

$$=2(-4+2)+3(-5-2)+1(-10-8)=-43$$

먼저 A의 소행렬식은 다음과 같다.

$$M_{11}=\begin{vmatrix}4 & 1\\ -2 & -1\end{vmatrix}=-2 \quad M_{12}=\begin{vmatrix}5 & 1\\ 2 & -1\end{vmatrix}=-7 \quad M_{13}=\begin{vmatrix}5 & 4\\ 2 & -2\end{vmatrix}=-18$$

$$M_{21}=\begin{vmatrix}-3 & 1\\ -2 & -1\end{vmatrix}=5 \quad M_{22}=\begin{vmatrix}2 & 1\\ 2 & -1\end{vmatrix}=-4 \quad M_{23}=\begin{vmatrix}2 & -3\\ 2 & -2\end{vmatrix}=2$$

$$M_{31}=\begin{vmatrix}-3 & 1\\ 4 & 1\end{vmatrix}=-7 \quad M_{32}=\begin{vmatrix}2 & 1\\ 5 & 1\end{vmatrix}=-3 \quad M_{33}=\begin{vmatrix}2 & -3\\ 5 & 4\end{vmatrix}=23$$

따라서 $(-1)^{i+j}$의 값에 따라 부호를 붙인 A의 여인수는 다음과 같다.

$$\begin{array}{lll} A_{11}=-2 & A_{12}=\ \ 7 & A_{13}=-18 \\ A_{21}=-5 & A_{22}=-4 & A_{23}=\ -2 \\ A_{31}=-7 & A_{32}=\ \ 3 & A_{33}=\ \ 23 \end{array}$$

여기서 잠깐 !! 수반행렬을 구할 경우에는 여인수 행렬에다 전치(transpose)하는 것을 잊지 말자. 또한 최종적으로 역행렬을 구할 때에는 수반행렬에다 $\dfrac{1}{\mathrm{Det}\,(A)}$을 곱하는 것도 잊지 말자.

그러므로 여인수 행렬 B는

$$B=\begin{bmatrix}-2 & 7 & -18\\ -5 & -4 & -2\\ -7 & 3 & 23\end{bmatrix}\text{이므로}$$

B의 전치행렬을 구하면 된다.

$$\mathrm{Adj}(A)=B^T=\begin{bmatrix}-2 & -5 & -7\\ 7 & -4 & 3\\ -18 & -2 & 23\end{bmatrix}\text{이고}$$

$$A^{-1}=\frac{1}{\mathrm{Det}(A)}\mathrm{Adj}(A)\text{이므로}$$

$$A^{-1}=\frac{1}{43}\begin{bmatrix}2 & 5 & 7\\ -7 & 4 & -3\\ 18 & 2 & -23\end{bmatrix}\text{이다.}$$

(2) $|A|=25,\ \mathrm{Adj}(A)=\begin{bmatrix} 2 & 7 & 1 \\ 19 & -21 & -3 \\ 15 & -10 & -5 \end{bmatrix},\ A^{-1}=\frac{1}{25}\begin{bmatrix} 2 & 7 & 1 \\ 19 & -21 & -3 \\ 15 & -10 & -5 \end{bmatrix}$인데

이 문제의 자세한 풀이는 각자 해 보기로 한다.

(3) $|A|=0,\ \mathrm{Adj}(A)=\begin{bmatrix} -16 & 7 & -1 \\ 32 & -14 & 2 \\ 16 & -7 & 1 \end{bmatrix},\ |A|=0$이므로

A^{-1}가 존재하지 않는다. ■

3.4 선형방정식의 해법

정리 ❸-14 $Ax=b$가 n개의 변수에 대한 n개의 방정식으로 이루어진 선형시스템이고, 행렬 A가 가역적이면 선형시스템은 유일한 해 $x=A^{-1}b$를 가진다.

〈그림 3.3〉 $Ax=b$와 $x=A^{-1}b$의 관계

예제 ❸-22 다음 선형시스템에서 역행렬을 이용하여 해를 구해 보자.

$$\begin{aligned} x \ + 2y + \ \ 3z &= 1 \\ x \ + 3y + \ \ 6z &= 3 \\ 2x + 6y + 13z &= 5 \end{aligned}$$

풀이 먼저 이 시스템에서 A 행렬과 b 행렬은 다음과 같다.

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 1 & 3 & 6 \\ 2 & 6 & 13 \end{bmatrix} \qquad b = \begin{bmatrix} 1 \\ 3 \\ 5 \end{bmatrix}$$

만약 역행렬 $A^{-1}=\begin{bmatrix} 3 & -8 & 3 \\ -1 & 7 & -3 \\ 0 & -2 & 1 \end{bmatrix}$가 구해졌다면 이 시스템의 유일한 해는

$$A^{-1}b=\begin{bmatrix} 3 & -8 & 3 \\ -1 & 7 & -3 \\ 0 & -2 & 1 \end{bmatrix}\begin{bmatrix} 1 \\ 3 \\ 5 \end{bmatrix}=\begin{bmatrix} -6 \\ 5 \\ -1 \end{bmatrix}$$이 된다.

따라서 최종적인 해는 $x=-6,\ y=5,\ z=-1$이다. ■

예제 3-23 다음 선형시스템의 해를 역행렬을 이용하여 구해 보자.

$$\begin{aligned} 3x_1 + 2x_2 + x_3 &= 5 \\ x_1 + x_2 + 2x_3 &= 1 \\ 2x_1 + x_2 + x_3 &= 2 \end{aligned}$$

풀이 주어진 선형시스템을 $Ax = b$의 형태로 쓸 수 있다. 여기서 $|A| \neq 0$이므로 A^{-1}가 존재하게 되며 역행렬을 구하면

$$A^{-1} = \frac{1}{2}\begin{bmatrix} -1 & -1 & 3 \\ 3 & 1 & -5 \\ -1 & 1 & 1 \end{bmatrix}$$ 이므로

이 선형시스템의 해는 $x = A^{-1}b$가 된다.

$$x=\begin{bmatrix}x_1\\x_2\\x_3\end{bmatrix}=\frac{1}{2}\begin{bmatrix}-1&-1&3\\3&1&-5\\-1&1&1\end{bmatrix}\begin{bmatrix}5\\1\\2\end{bmatrix}=\frac{1}{2}\begin{bmatrix}0\\6\\-2\end{bmatrix}=\begin{bmatrix}0\\3\\-1\end{bmatrix}\text{이다.}$$

따라서 최종적인 해는 $x_1=0$, $x_2=3$, $x_3=-1$이다. ■

정리 ❸-15 크래머의 규칙(Cramer's rule)은 다음과 같다.

만약 $A=[a_{ij}]$가 가역적이고

$$b=\begin{bmatrix} b_1 \\ \vdots \\ b_n \end{bmatrix}$$ 라면

$Ax=b$의 해집합은 다음과 같다.

$$x=\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$

x_1의 값을 구하기 위해서는 행렬 A의 첫 번째 열 대신에 b가 대체된다.

$$x_1 = \frac{1}{\mathrm{Det}\,(A)}\,\mathrm{Det}\begin{bmatrix} b_1 & a_{12} & \cdots & a_{1n} \\ b_2 & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \cdots & \vdots \\ b_n & a_{n2} & \cdots & a_{nn} \end{bmatrix}$$

x_2의 경우에는 행렬 A의 두 번째 열 대신에 b가 대체된다.

$$x_2 = \frac{1}{\mathrm{Det}\,(A)}\,\mathrm{Det}\begin{bmatrix} a_{11} & b_1 & \cdots & a_{1n} \\ a_{21} & b_2 & \cdots & a_{2n} \\ \vdots & \vdots & \cdots & \vdots \\ a_{n1} & b_n & \cdots & a_{nn} \end{bmatrix}$$

이와 같은 방식으로 x_n은 A의 n번째 열 대신에 b가 대체되어 만들어진다.

$$x_n = \frac{1}{\mathrm{Det}\,(A)}\,\mathrm{Det}\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1,n-1} & b_1 \\ a_{21} & a_{22} & \cdots & a_{2,n-1} & b_2 \\ \vdots & \vdots & \cdots & \vdots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{n,n-1} & b_n \end{bmatrix}$$

C program

예제 3-24 크래머의 규칙을 이용하여 다음 선형시스템의 해를 구해 보자.

$$\begin{aligned} 3x_1 - 2x_2 &= 6 \\ -5x_1 + 4x_2 &= 8 \end{aligned}$$

풀이 주어진 시스템을 $Ax = b$로 보고 크래머의 규칙을 적용한다.

$$A = \begin{bmatrix} 3 & -2 \\ -5 & 4 \end{bmatrix}$$

이고 A의 행렬식 $\text{Det}(A) = 2$이므로 이 시스템은 유일한 해를 가진다는 것을 알 수 있다. 주어진 선형시스템의 해를 크래머의 규칙에 따라 구하면 다음과 같다.

$$x_1 = \frac{\begin{vmatrix} 6 & -2 \\ 8 & 4 \end{vmatrix}}{\mathrm{Det}\,(A)} = \frac{24+16}{2} = 20$$

$$x_2 = \frac{\begin{vmatrix} 3 & 6 \\ -5 & 8 \end{vmatrix}}{\mathrm{Det}\,(A)} = \frac{24+30}{2} = 27 \quad \blacksquare$$

예제 ❸-25 다음과 같은 선형시스템을 크래머의 규칙을 이용하여 풀어 보자.

$$\begin{aligned} x_1 - x_2 + x_3 &= 0 \\ x_1 + x_2 - 2x_3 &= 1 \\ x_1 + 2x_2 + x_3 &= 6 \end{aligned}$$

풀이 먼저 $|A|$를 구하면

$|A| = \begin{vmatrix} 1 & -1 & 1 \\ 1 & 1 & -2 \\ 1 & 2 & 1 \end{vmatrix} = 9$이므로 주어진 선형시스템은 유일한 해를 가진다.

따라서

$$x_1 = \frac{\begin{vmatrix} 0 & -1 & 1 \\ 1 & 1 & -2 \\ 6 & 2 & 1 \end{vmatrix}}{9} = 1, \quad x_2 = \frac{\begin{vmatrix} 1 & 0 & 1 \\ 1 & 1 & -2 \\ 1 & 6 & 1 \end{vmatrix}}{9} = 2,$$

$$x_3 = \frac{\begin{vmatrix} 1 & -1 & 0 \\ 1 & 1 & 1 \\ 1 & 2 & 6 \end{vmatrix}}{9} = 1$$
이다. ■

C program

예제 ❸-26

다음과 같은 선형시스템을 크래머의 규칙에 따라 해를 구해 보자.

$$
\begin{aligned}
2x_1 + 3x_2 - x_3 &= 2 \\
x_1 + 2x_2 + x_3 &= -1 \\
2x_1 + x_2 - 6x_3 &= 4
\end{aligned}
$$

여기서 위의 식을 행렬의 형태로 각각 바꾸면

$$
A = \begin{bmatrix} 2 & 3 & -1 \\ 1 & 2 & 1 \\ 2 & 1 & -6 \end{bmatrix}, \quad b = \begin{bmatrix} 2 \\ -1 \\ 4 \end{bmatrix}
$$

와 같다. 먼저 $\mathrm{Det}(A)$를 구하면 $\mathrm{Det}(A) = 1$이 된다. 따라서 크래머의 규칙을 적용하면

$$x_1 = \mathrm{Det}\begin{bmatrix} 2 & 3 & -1 \\ -1 & 2 & 1 \\ 4 & 1 & -6 \end{bmatrix} = -23$$

$$x_2 = \mathrm{Det}\begin{bmatrix} 2 & 2 & -1 \\ 1 & -1 & 1 \\ 2 & 4 & -6 \end{bmatrix} = 14$$

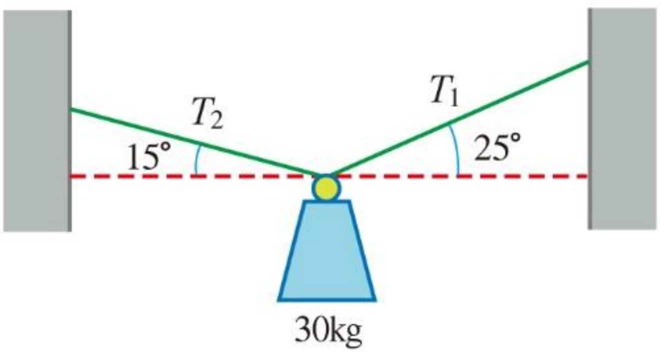
$$x_3 = \mathrm{Det}\begin{bmatrix} 2 & 3 & 2 \\ 1 & 2 & -1 \\ 2 & 1 & 4 \end{bmatrix} = -6$$

을 얻을 수 있다. ■

예제 ❸-27 〈그림 3.4〉에서 전선에 있는 장력의 크기 T_1과 T_2는 방정식

$$(\cos 25°)T_1 - (\cos 15°)T_2 = 0$$
$$(\sin 25°)T_1 + (\sin 15°)T_2 = 30$$

을 만족한다고 할 때 크래머의 규칙을 이용하여 T_1과 T_2의 해를 구해 보자.



〈그림 3.4〉 전선의 장력

풀이 먼저 행렬식을 구하면

$$\begin{aligned}\mathrm{Det}(A) &= \cos 25^\circ \cdot \sin 15^\circ + \sin 25^\circ \cdot \cos 15^\circ \\ &= \sin(25+15)^\circ \\ &= \sin 40^\circ\end{aligned}$$

다음으로 크래머의 규칙에 따라 대입하면 다음과 같은 값을 구할 수 있다.

$$T_1 = \frac{\begin{vmatrix} 0 & -\cos 15^\circ \\ 30 & \sin 15^\circ \end{vmatrix}}{\sin 40^\circ} = \frac{30\cos 15^\circ}{\sin 40^\circ} = 45.08\mathrm{kg}$$

$$T_2 = \frac{\begin{vmatrix} \cos 25^\circ & 0 \\ \sin 25^\circ & 30 \end{vmatrix}}{\sin 40^\circ} = \frac{30\cos 25^\circ}{\sin 40^\circ} = 42.30\mathrm{kg} \quad \blacksquare$$

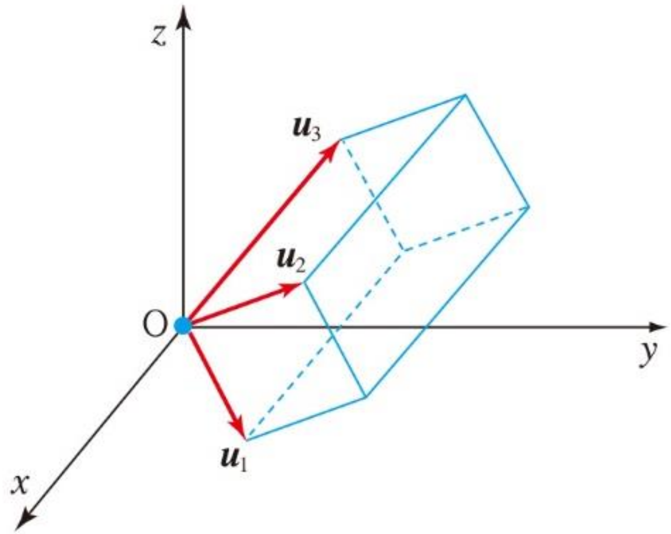
예제 3-28 R^3상의 세 개의 벡터가 주어졌을 때, 평행육면체 S의 체적인 $V(S)$를 구해 보자.

$$\boldsymbol{u}_1=(1,\ 1,\ 0),\qquad \boldsymbol{u}_2=(1,\ 1,\ 1),\qquad \boldsymbol{u}_3=(0,\ 2,\ 3)$$

풀이 각각의 행이 $\boldsymbol{u}_1$, $\boldsymbol{u}_2$, $\boldsymbol{u}_3$인 행렬식의 값을 구하면 〈그림 3.5〉와 같은 평행육면체의 체적이 된다.

$$\begin{vmatrix} 1 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 2 & 3 \end{vmatrix} = 3+0+0-0-2-3=-2$$

따라서 $V(S)=|-2|=2$이다. ■



〈그림 3.5〉 행렬식에 의해 결정되는 평행육면체의 체적

```c
#include <stdio.h>
#include <stdlib.h>
#include <math.h>

#define N 4                               // 행과 열의 최대값

double det(double (*A)[N], int n);        // 행렬식을 구하는 자기호출 함수

// main 함수
void main()
{

    int i = 0, j = 0;                     // 루프를 수행하기 위한 변수 선언
    int n;                                // 입력받을 행렬 값의 변수
    double A[N][N] = {0,};                // 입력받을 행렬의 2차원 배열
```

```
printf(" ********************************************************\n");
printf(" **                                                    **\n");
printf(" **                  행렬식 계산 프로그램               **\n");
printf(" **                                                    **\n");
printf(" ******************************************************\n\n");

// 행렬의 크기 값 입력
printf(" 행렬의 크기 입력: ");
scanf("%d", &n);

printf("\n");
printf(" 행렬의 값을 입력하세요. \n");
```

3.5 컴퓨터 프로그램에 의한 연산

```
// 각각의 행렬 값 입력
for(i = 0; i < n; i++)
{
    for(j = 0; j < n; j++)
    {
        printf(" %d X %d 행렬의 값을 입력하세요: ", i+1, j+1);
        scanf("%lf", &A[i][j]);
    }
}

// 입력한 행렬 값 출력
printf("\n 입력한 행렬 A = \n");
for(i = 0;i < n;i++)
{
    printf("\t|\t");
    for(j = 0;j < n;j++)
```

```
        {
            printf("%.f\t", A[i][j]);
        }
        printf("|\n");
    }
    printf("\n");

    // 행렬식 값을 출력
    printf(" 입력한 행렬의 행렬식 값\n");
    printf(" Det(A) = %.f\n\n", det(A,n));

}
```

3.5 컴퓨터 프로그램에 의한 연산

```
// 행렬식 값 계산 함수
double det(double (*A)[N], int n)
{
    int i, j, k, y;               // 루프를 수행하기 위한 변수 선언
    double det_a = 0;             // 결과 값 저장 변수
    double temp[N][N];            // 행렬식 값 임시 저장 배열

    // 알고리즘
    if(n != 1)
    {
        for(i = 0; i < n; i++)
        {
            for(j = 1; j < n; j++)
            {
                y = 0;
```

```
                for(k = 0; k < n; k++)
                {
                    if(k != i)
                    {
                        temp[j-1][y++] = *(A[0]+j*N+k);
// 소행렬식을 구하기 위해 각 행렬의 원소들을 재배치한다.
                    }
                }
            }
            det_a = det_a + *(A[0]+i) * (pow(-1,i)) * det(temp,n-1);
// 소행렬식들의 전체 합을 구한다.
        }
        return det_a;
    }
    else
    {
        return *A[0];
    }
}
```

3.5 컴퓨터 프로그램에 의한 연산

실습 ❸−1

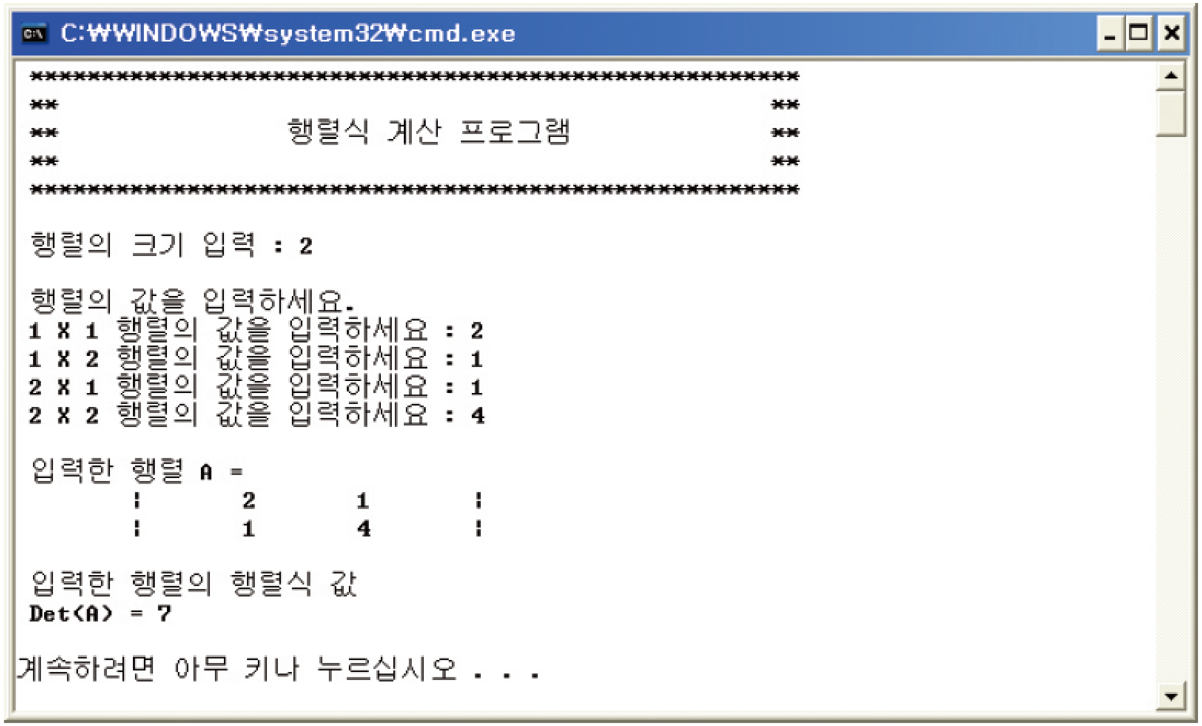
예제 ③−1 | C program, MATLAB

다음 행렬 A, B의 행렬식을 각각 구해 보자.

$$A=\begin{bmatrix}2 & 1\\1 & 4\end{bmatrix},\quad B=\begin{bmatrix}-2 & -3\\4 & 5\end{bmatrix}$$

풀이 A의 행렬식은 $2\cdot4-1\cdot1=7$이다.

또한 B의 행렬식은 $(-2)\cdot5-(-3)\cdot4=-10+12=2$이다. ■



```
C:₩WINDOWS₩system32₩cmd.exe
**********************************************************
**                                                      **
**                  행렬식 계산 프로그램                 **
**                                                      **
**********************************************************

행렬의 크기 입력 : 2

행렬의 값을 입력하세요.
1 X 1 행렬의 값을 입력하세요 : 2
1 X 2 행렬의 값을 입력하세요 : 1
2 X 1 행렬의 값을 입력하세요 : 1
2 X 2 행렬의 값을 입력하세요 : 4

입력한 행렬 A =
        |       2       1       |
        |       1       4       |

입력한 행렬의 행렬식 값
Det(A) = 7

계속하려면 아무 키나 누르십시오 . . .
```

컴퓨터 프로그램에 의한 연산

```
C:₩WINDOWS₩system32₩cmd.exe
*******************************************************
**                                                   **
**                 행렬식 계산 프로그램              **
**                                                   **
*******************************************************

행렬의 크기 입력 : 2

행렬의 값을 입력하세요.
1 X 1 행렬의 값을 입력하세요 : -2
1 X 2 행렬의 값을 입력하세요 : -3
2 X 1 행렬의 값을 입력하세요 : 4
2 X 2 행렬의 값을 입력하세요 : 5

입력한 행렬 A =
        |       -2       -3       |
        |        4        5       |

입력한 행렬의 행렬식 값
Det(A) = 2

계속하려면 아무 키나 누르십시오 . . .
```

실습 ❸−2

예제 ③−3 C program, MATLAB

다음 행렬 A의 행렬식을 제1행에 대한 여인수들로 전개하여 구해 보자.

$$A=\begin{bmatrix}2 & 4 & 7\\ 6 & 0 & 3\\ 1 & 5 & 3\end{bmatrix}$$

풀이

$$\mathrm{Det}(A)=2(-1)^{1+1}\begin{vmatrix}0 & 3\\ 5 & 3\end{vmatrix}+4(-1)^{1+2}\begin{vmatrix}6 & 3\\ 1 & 3\end{vmatrix}+7(-1)^{1+3}\begin{vmatrix}6 & 0\\ 1 & 5\end{vmatrix}$$

$$=2(0\cdot 3-3\cdot 5)-4(6\cdot 3-3\cdot 1)+7(6\cdot 5-0\cdot 1)$$

$$=120 \ ■$$

```
C:WWINDOWSWsystem32Wcmd.exe
*******************************************************
**                                                   **
**                 행렬식 계산 프로그램              **
**                                                   **
*******************************************************

행렬의 크기 입력 : 3

행렬의 값을 입력하세요.
1 X 1 행렬의 값을 입력하세요 : 2
1 X 2 행렬의 값을 입력하세요 : 4
1 X 3 행렬의 값을 입력하세요 : 7
2 X 1 행렬의 값을 입력하세요 : 6
2 X 2 행렬의 값을 입력하세요 : 0
2 X 3 행렬의 값을 입력하세요 : 3
3 X 1 행렬의 값을 입력하세요 : 1
3 X 2 행렬의 값을 입력하세요 : 5
3 X 3 행렬의 값을 입력하세요 : 3

입력한 행렬 A =
       |       2       4       7       |
       |       6       0       3       |
       |       1       5       3       |

입력한 행렬의 행렬식 값
Det(A) = 120

계속하려면 아무 키나 누르십시오 . . .
```

실습 ❸-3

예제 ③-9 C program, MATLAB

행렬식의 성질을 이용하여 다음 행렬식의 값을 구해 보자.

$$|A| = \begin{vmatrix} 2 & 1 & 3 & 4 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \\ 4 & 2 & 1 & -1 \end{vmatrix}$$

풀이 $|A| = 1 \times 1 \times (-5) \times 2 = -10$ ■

3.5 컴퓨터 프로그램에 의한 연산

```
C:\WINDOWS\system32\cmd.exe
*******************************************************
**                                                   **
**                 행렬식 계산 프로그램                **
**                                                   **
*******************************************************

행렬의 크기 입력 : 4

행렬의 값을 입력하세요.
1 X 1 행렬의 값을 입력하세요 : 2
1 X 2 행렬의 값을 입력하세요 : 1
1 X 3 행렬의 값을 입력하세요 : 3
1 X 4 행렬의 값을 입력하세요 : 4
2 X 1 행렬의 값을 입력하세요 : 0
2 X 2 행렬의 값을 입력하세요 : 1
2 X 3 행렬의 값을 입력하세요 : 1
2 X 4 행렬의 값을 입력하세요 : 0
3 X 1 행렬의 값을 입력하세요 : 1
3 X 2 행렬의 값을 입력하세요 : 0
3 X 3 행렬의 값을 입력하세요 : 1
3 X 4 행렬의 값을 입력하세요 : 1
4 X 1 행렬의 값을 입력하세요 : 4
4 X 2 행렬의 값을 입력하세요 : 2
4 X 3 행렬의 값을 입력하세요 : 1
4 X 4 행렬의 값을 입력하세요 : -1

입력한 행렬 A =
       |       2       1       3       4       |
       |       0       1       1       0       |
       |       1       0       1       1       |
       |       4       2       1       -1      |

입력한 행렬의 행렬식 값
Det(A) = -10

계속하려면 아무 키나 누르십시오 . . .
```

3.5 컴퓨터 프로그램에 의한 연산

```c
#include <stdio.h>
#include <stdlib.h>
#include <math.h>

typedef struct _MATRIX
{
    double **m_data;
    int m_size;
}MATRIX;

void initMatrix(MATRIX *A, int n);                      // 동적배열 생성함수
void deleteMatrix(MATRIX *A);                           // 동적배열 해제함수
void inputMatrix(MATRIX *A);                            // 행렬 값 입력함수
void printMatrix(MATRIX matrix);                        // 행렬 값 출력함수
double determinant(MATRIX matrix);                      // 행렬식 계산함수
MATRIX transpose(MATRIX matrix);                        // 전치행렬 계산함수
MATRIX minorMatrix(MATRIX matrix, int col, int row);    // 소행렬식 계산함수
MATRIX cofactorMatrix(MATRIX matrix);                   // 여인자행렬 계산함수
```

```c
MATRIX adjoint(MATRIX matrix);                   // 수반행렬 계산함수
MATRIX inverseMatrix(MATRIX matrix);             // 역행렬 계산함수

int main (void)
{
   MATRIX matrix;
   MATRIX inverse;

   int n;
   double det = 0;

   printf(" *********************************************************\n");
   printf(" **                                                     **\n");
   printf(" **         수반행렬을 이용한 역행렬 계산 프로그램         **\n");
   printf(" **                                                     **\n");
   printf(" *******************************************************\n\n");
```

```c
// 행렬의 크기 값을 입력한다.
printf(" 행렬의 크기입력: ");
scanf("%d", &n);

// matrix 구조체와 inverse 구조체를 동적배열로 행렬을 생성한다.
initMatrix(&matrix, n);
initMatrix(&inverse, n);

// matrix 구조체에 사용자의 입력 값을 받는다.
inputMatrix(&matrix);

// matrix 구조체의 행렬식 값을 계산하여 det 변수에 저장한다.
det = determinant(matrix);

// matrix 구조체의 역행렬 값을 계산하여 inverse 구조체에 저장한다.
inverse = inverseMatrix(matrix);
```

```
    // 입력한 행렬의 행렬식 값을 출력한다.
    printf("\n 입력한 행렬의 행렬식 값\n");
    printf(" Det(A) = %.3lf\n\n", det);

    // 입력한 행렬의 역행렬을 출력한다.
    printf(" 역행렬= \n");
    printMatrix(inverse);
    printf("\n");

    // matrix 구조체와 inverse 구조체에 할당된 동적배열을 해제한다.
    deleteMatrix(&matrix);
    deleteMatrix(&inverse);

    return 0;
}

// 입력의 크기에 따라 동적배열을 생성한다.
void initMatrix (MATRIX *A , int n)
{
```

```
    int i = 0;

    A->m_data = (double **)malloc(sizeof(double*) * n);

    for( i = 0; i < n; i++ )
    {
        A->m_data[i] = (double*)malloc(sizeof(double) * n);
    }

    A->m_size = n;
}

// 사용하고 난 후 다른 입력을 위해 동적배열을 해제한다.
void deleteMatrix (MATRIX *A)
{
    int i = 0;
```

```
    for( i = 0 ; i < A->m_size ; i++ )
    {
        free(A->m_data[i]);
    }
    free (A->m_data);
}

// 행렬 값을 입력한다.
void inputMatrix(MATRIX* A)
{
    int i = 0, j = 0;
    double input = 0;

    printf("\n");
    printf(" 행렬의 값을 입력하세요. \n");

    // 각각의 행렬 값을 입력한다.
    for( i = 0; i < A->m_size; i++ )
    {
```

```
        for( j = 0; j < A->m_size; j++ )
        {
            fflush(stdin);
            printf(" %d X %d 행렬의 값을 입력하세요: " , i+1, j+1);
            scanf("%lf", &input);
            A->m_data[i][j] = input;
        }
    }
}

// 행렬 값을 출력한다.
void printMatrix(MATRIX matrix)
{
    int i = 0, j = 0;

    for( i = 0; i < matrix.m_size; i++ )
```

```
    {
        printf("\t|\t");
        for( j = 0; j < matrix.m_size; j++)
        {
            printf("%.3lf \t", matrix.m_data[i][j]);
        }
        printf("|\n");
    }
}

// 행렬식을 계산한다(어떤 크기의 행렬 값도 계산한다).
double determinant(MATRIX matrix)
{
    int i = 0;
    double det = 0;
    int sign = 1;
```

```
    // 2x2일 경우 행렬식을 구하는 과정
    if( matrix.m_size == 2 )
    {
        det = matrix.m_data[0][0] * matrix.m_data[1][1] -matrix.m_data[1][0] *
matrix.m_data[0][1];
        return det;
    }

    // 3x3 이상인 경우 소행렬식을 이용하여 행렬식을 구하는 과정
    for( i = 0 ; i < matrix.m_size ; i++ )
    {
        MATRIX minor;
        initMatrix(&minor, matrix.m_size);

        minor = minorMatrix(matrix, 0, i);
        det = det + sign * matrix.m_data[i][0] * determinant(minor);
        sign = sign * -1;
    }
```

```
    return det;
}

// 행과 열을 서로 바꾸어 전치행렬로 만든다.
MATRIX transpose(MATRIX matrix)
{
    MATRIX Result;
    int i, j;

    initMatrix( &Result , matrix.m_size);

    for( i = 0 ; i < matrix.m_size ; i++ )
    {
        for( j = 0 ; j < matrix.m_size ; j++ )
        {
            Result.m_data[i][j] = matrix.m_data[j][i];
        }
    }

    return Result;
```

```
}

// 행과 열의 크기를 각각 1씩 줄인 소행렬식의 값을 구한다.
MATRIX minorMatrix(MATRIX matrix, int col, int row)
{
    MATRIX Result;
    int i, j;
    int rowIndex = 0;
    int colIndex = 0;

    initMatrix( &Result , matrix.m_size - 1 );

    for( i = 0 ; i < matrix.m_size ; i++ )
    {
        for( j = 0 ; j < matrix.m_size ; j++ )
        {
            if( i != row && j != col )
```

```
                {
                    Result.m_data[rowIndex][colIndex] = matrix.m_data[i][j];
                    colIndex++;
                }
            }

            if( i != row && j != col )
            {
                colIndex = 0;
                rowIndex++;
            }
        }

        return Result;
    }

    // 여인수행렬을 계산하여 만든다.
    MATRIX cofactorMatrix(MATRIX matrix)
    {
```

```
    MATRIX Result;
    int i, j;

    initMatrix(&Result, matrix.m_size);

    for( i = 0 ; i < matrix.m_size ; i++ )
    {
        for( j = 0 ; j < matrix.m_size ; j++ )
        {
            Result.m_data[j][i] = determinant(minorMatrix(matrix , i , j));
        }
    }

    return Result;
}
```

```
// 여인수행렬을 전치하여 수반행렬을 만든다.
MATRIX adjoint(MATRIX matrix)
{
   MATRIX Result;
   MATRIX confactor;
   MATRIX transposed;
   int i, j;
   double ipow = 1;

   initMatrix(&Result , matrix.m_size);
   initMatrix(&confactor , matrix.m_size);
   initMatrix(&transposed , matrix.m_size);

   confactor = cofactorMatrix(matrix);
   transposed = transpose(confactor);

   for( i = 0 ; i < matrix.m_size ; i++ )
   {
      for( j = 0 ; j < matrix.m_size ; j++ )
      {
```

```
            ipow = pow(-1, (i+j));
            Result.m_data[i][j] = ipow *transposed.m_data[i][j];
        }
    }
    deleteMatrix(&confactor);
    deleteMatrix(&transposed);

    return Result;
}

// 수반행렬을 이용한 최종적인 역행렬을 생성한다.
MATRIX inverseMatrix(MATRIX matrix)
{
    MATRIX Result;
    MATRIX temp;
    double det;
    int i, j;
```

```
det = determinant(matrix);

initMatrix(&temp, matrix.m_size);
initMatrix(&Result, matrix.m_size);

// 2x2일 경우 역행렬을 구하는 과정
if(matrix.m_size == 2)
{
    temp.m_data[0][0] = matrix.m_data[1][1];
    temp.m_data[1][1] = matrix.m_data[0][0];
    temp.m_data[0][1] = -(matrix.m_data[0][1]);
    temp.m_data[1][0] = -(matrix.m_data[1][0]);

    for(i = 0; i < matrix.m_size; i++)
    {
        for(j = 0; j < matrix.m_size; j++)
        {
            Result.m_data[i][j] = (double)((double)1/det) * temp.m_data[i][j];
```

```
            }
        }
        deleteMatrix(&temp);

        return Result;
    }

    // 3x3 이상인 일반적인 경우 역행렬을 구하는 과정
    temp = adjoint(matrix);

    for( i = 0 ; i < matrix.m_size ; i++ )
    {
        for( j = 0 ; j < matrix.m_size ; j++ )
        {
            Result.m_data[i][j] = (double)((double)1/det) * temp.m_data[i][j];
        }
    }
    deleteMatrix(&temp);

    return Result;
}
```

예제 ③-20 C program, MATLAB

실습 ③-4

$A=\begin{bmatrix} 3 & 1 \\ 4 & 2 \end{bmatrix}$일 때 수반행렬을 이용하여 A의 역행렬을 구해 보자.

풀이 $|A|=\begin{vmatrix} 3 & 1 \\ 4 & 2 \end{vmatrix}=2$이므로 가역적이다.

따라서 $A^{-1}=\frac{1}{2}\begin{bmatrix} 2 & -1 \\ -4 & 3 \end{bmatrix}$이 된다. ■

```
C:\WINDOWS\system32\cmd.exe
*******************************************************
**                                                   **
**      수반행렬을 이용한 역행렬 계산 프로그램          **
**                                                   **
*******************************************************

행렬의 크기 입력 : 2

행렬의 값을 입력하세요.
1 X 1 행렬의 값을 입력하세요 : 3
1 X 2 행렬의 값을 입력하세요 : 1
2 X 1 행렬의 값을 입력하세요 : 4
2 X 2 행렬의 값을 입력하세요 : 2

입력한 행렬의 행렬식 값
Det(A) = 2.000

역행렬 =
       |       1.000   -0.500  |
       |       -2.000  1.500   |

계속하려면 아무 키나 누르십시오 . . .
```

실습 ❸-5

예제 ❸-21 C program, MATLAB

다음에서 행렬식과 Adj(A)를 각각 구하고 역행렬이 존재하면 구해 보자.

$$(2)\ A=\begin{bmatrix} 3 & 1 & 0 \\ 2 & -1 & 1 \\ 5 & 5 & -7 \end{bmatrix}$$

풀이 먼저 행렬식을 구하고 Adj(A)를 만든 후 역행렬을 구한다.

$$(2)\ |A|=25,\quad \mathrm{Adj}(A)=\begin{bmatrix} 2 & 7 & 1 \\ 19 & -21 & -3 \\ 15 & -10 & -5 \end{bmatrix},$$

$$A^{-1}=\frac{1}{25}\begin{bmatrix} 2 & 7 & 1 \\ 19 & -21 & -3 \\ 15 & -10 & -5 \end{bmatrix}$$ ■

3.5 컴퓨터 프로그램에 의한 연산

C:\WINDOWS\system32\cmd.exe

```
*******************************************************
**                                                   **
**       수반행렬을 이용한 역행렬 계산 프로그램       **
**                                                   **
*******************************************************

행렬의 크기 입력 : 3

행렬의 값을 입력하세요.
1 X 1 행렬의 값을 입력하세요 : 3
1 X 2 행렬의 값을 입력하세요 : 1
1 X 3 행렬의 값을 입력하세요 : 0
2 X 1 행렬의 값을 입력하세요 : 2
2 X 2 행렬의 값을 입력하세요 : -1
2 X 3 행렬의 값을 입력하세요 : 1
3 X 1 행렬의 값을 입력하세요 : 5
3 X 2 행렬의 값을 입력하세요 : 5
3 X 3 행렬의 값을 입력하세요 : -7

입력한 행렬의 행렬식 값
Det(A) = 25.000

역행렬 =
       |       0.080   0.280   0.040   |
       |       0.760  -0.840  -0.120   |
       |       0.600  -0.400  -0.200   |

계속하려면 아무 키나 누르십시오 . . .
```

3.5 컴퓨터 프로그램에 의한 연산

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>

#define N 5                              // 행과 열의 최대값

double det(double (*A)[N], int n);       // 행렬식을 구하는 자기호출함수
void cramer(double (*A)[N], double C[N], int n, int c);
// 크래머의 규칙을 이용한 선형방정식 결과 값 계산함수

// main 함수
void main()
{
   int i = 0, j = 0;                     // 루프를 수행하기 위한 변수 선언
   int n;                                // 입력받을 행렬 값의 변수
   double A[N][N] = {0,};                // 입력받을 행렬의 2차원 배열(방정식)
```

```
double C[N] = {0,};                    // 입력받을 행렬의 1차원 배열(결과 값)

printf(" ********************************************************\n");
printf(" **                                                    **\n");
printf(" **     크래머의 규칙을 이용한 선형방정식 계산 프로그램     **\n");
printf(" **                                                    **\n");
printf(" ******************************************************\n\n");

// 행렬의 크기 값을 입력한다.
printf(" 선형방정식의 최대 차수를 입력하세요: ");
scanf("%d", &n);

printf("\n");

// 선형방정식의 수식을 입력한다.
printf(" 선형방정식의 수식을 입력하세요. \n");
```

```
for(i = 0; i < n; i++)
{
    for(j = 0; j < n; j++)
    {
        printf(" %d 번째 선형방정식 x%d 의 값: ", i+1, j+1);
        scanf("%lf", &A[i][j]);
    }
}
printf("\n");

// 선형방정식의 결과 값을 입력한다.
printf(" 선형방정식의 결과 값을 입력하세요. \n");

for(i = 0; i < n; i++)
{
    printf(" %d 번째 선형방정식의 결과 값: ", i+1);
    scanf("%lf", &C[i]);
}
```

```
// 입력한 행렬 값을 출력한다.
printf("\n 입력한 선형방정식의 행렬= \n\n");
for(i = 0; i < n; i++)
{
    printf(" |\t");
    for(j = 0; j < n; j++)
    {
        printf(" %.f x%d\t", A[i][j], j+1);
    }
    printf("=\t");
    printf("%.f\t", C[i]);
    printf("|\n");
}
printf("\n");

// 입력한 행렬 A(선형방정식)의 행렬식 값을 출력한다.
printf(" 입력한 선형방정식의 행렬식 값\n");
printf(" Det(A) = %.f\n\n", det(A, n));
```

```
    // 입력한 행렬 A(선형방정식)에 대한 해답을 출력한다.
    printf(" 입력한 선형방정식의 해답\n");

    for(i = 1; i < n+1; i++)
    {
        cramer(A, C, n, i);
    }
    printf("\n\n");

}
// 행렬식 값을 구하는 자기호출함수
double det(double (*A)[N], int n)
{

    int i, j, k, y;             // 루프를 수행하기 위한 변수 선언
```

```
    double det_a = 0;            // 결과 값 저장 변수
    double temp[N][N];           // 행렬식 값 임시 저장 배열

    if(n != 1)
    {
        for(i = 0; i < n; i++)
        {
            for(j = 1; j < n; j++)
            {
                y = 0;

                for(k = 0; k < n; k++)
                {
                    if(k != i)
                    {
                        temp[j-1][y++] = *(A[0]+j*N+k);
// 소행렬식을 구하기 위해 각 행렬의 원소들을 재배치한다.
                    }
                }
            }
```

```
            det_a = det_a + *(A[0]+i) * (pow(-1,i)) * det(temp,n-1);
// 소행렬식들의 전체합을 구한다.

        } return det_a;

    } else return *A[0];

}

// 크래머의 규칙을 이용한 선형방정식 결과 값 계산함수
void cramer(double (*A)[N], double C[N], int n, int c)
{
    int i = 0, j = 0;             // 루프를 수행하기 위한 변수 선언

    double temp[N][N] = {0,};
```

```
// 입력받은 행렬 A의 값을 임시 배열에 저장한다.
for(i = 0; i < n; i++)
{
    for(j = 0; j < n; j++)
    {
        temp[i][j] = A[i][j];
    }
}

// 행렬 A의 값을 넘겨받은 임시 배열 temp의 각 열에 C의 값을 대입한다.
for(i = 0; i < n; i++)
{
    for(j = c-1; j < c; j++)
    {
        temp[i][j] = C[i];
    }
}
```

3.5 컴퓨터 프로그램에 의한 연산

```
    // 선형방정식의 해를 출력한다(부분행렬의 행렬식/전체행렬의 행렬식).
    printf(" x%d = %.f \t", c, (det(temp, n) / det(A,n)) );


}
```

실습 ❸-6

예제 ❸-24

C program

크래머의 규칙을 이용하여 다음 선형시스템의 해를 구해 보자.

$$
\begin{aligned}
3x_1 - 2x_2 &= 6 \\
-5x_1 + 4x_2 &= 8
\end{aligned}
$$

풀이 $x_1 = 20,\ x_2 = 27$ ■

3.5 컴퓨터 프로그램에 의한 연산

C:\WINDOWS\system32\cmd.exe

```
************************************************************
**                                                        **
**  크래머의 규칙을 이용한 선형방정식 계산 프로그램 **
**                                                        **
************************************************************

선형방정식의 최대 차수를 입력하세요 : 2

선형방정식의 수식을 입력하세요.
1 번째 선형방정식 x1 의 값 : 3
1 번째 선형방정식 x2 의 값 : -2
2 번째 선형방정식 x1 의 값 : -5
2 번째 선형방정식 x2 의 값 : 4

선형방정식의 결과값을 입력하세요.
1 번째 선형방정식의 결과값 : 6
2 번째 선형방정식의 결과값 : 8

입력한 선형방정식의 행렬 =

|        3 x1     -2 x2    =        6        |
|       -5 x1     4 x2     =        8        |

입력한 선형방정식의 행렬식 값
Det(A) = 2

입력한 선형방정식의 해답
x1 = 20          x2 = 27

계속하려면 아무 키나 누르십시오 . . .
```

실습 ❸-7

예제 ❸-26 C program

다음과 같은 선형시스템에서 크래머의 규칙에 따라 해를 구해 보자.

$$\begin{aligned} 2x_1 + 3x_2 - x_3 &= 2 \\ x_1 + 2x_2 + x_3 &= -1 \\ 2x_1 + x_2 - 6x_3 &= 4 \end{aligned}$$

여기서 이것을 행렬의 형태로 바꾸면

$$A = \begin{bmatrix} 2 & 3 & -1 \\ 1 & 2 & 1 \\ 2 & 1 & -6 \end{bmatrix}, \quad b = \begin{bmatrix} 2 \\ -1 \\ 4 \end{bmatrix}$$

풀이 $x_1 = -23, \ x_2 = 14, \ x_3 = -6$ ■

컴퓨터 프로그램에 의한 연산

```
C:₩WINDOWS₩system32₩cmd.exe
*****************************************************
**                                                 **
**  크래머의 규칙을 이용한 선형방정식 계산 프로그램 **
**                                                 **
*****************************************************

선형방정식의 최대 차수를 입력하세요 : 3

선형방정식의 수식을 입력하세요.
1 번째 선형방정식 x1 의 값 : 2
1 번째 선형방정식 x2 의 값 : 3
1 번째 선형방정식 x3 의 값 : -1
2 번째 선형방정식 x1 의 값 : 1
2 번째 선형방정식 x2 의 값 : 2
2 번째 선형방정식 x3 의 값 : 1
3 번째 선형방정식 x1 의 값 : 2
3 번째 선형방정식 x2 의 값 : 1
3 번째 선형방정식 x3 의 값 : -6

선형방정식의 결과값을 입력하세요.
1 번째 선형방정식의 결과값 : 2
2 번째 선형방정식의 결과값 : -1
3 번째 선형방정식의 결과값 : 4

입력한 선형방정식의 행렬 =

|       2 x1     3 x2    -1 x3   =       2        |
|       1 x1     2 x2     1 x3   =      -1        |
|       2 x1     1 x2    -6 x3   =       4        |

입력한 선형방정식의 행렬식 값
Det(A) = 1

입력한 선형방정식의 해답
x1 = -23          x2 = 14          x3 = -6

계속하려면 아무 키나 누르십시오 . . .
```

3.5 컴퓨터 프로그램에 의한 연산

예제 ③-1 C program, MATLAB

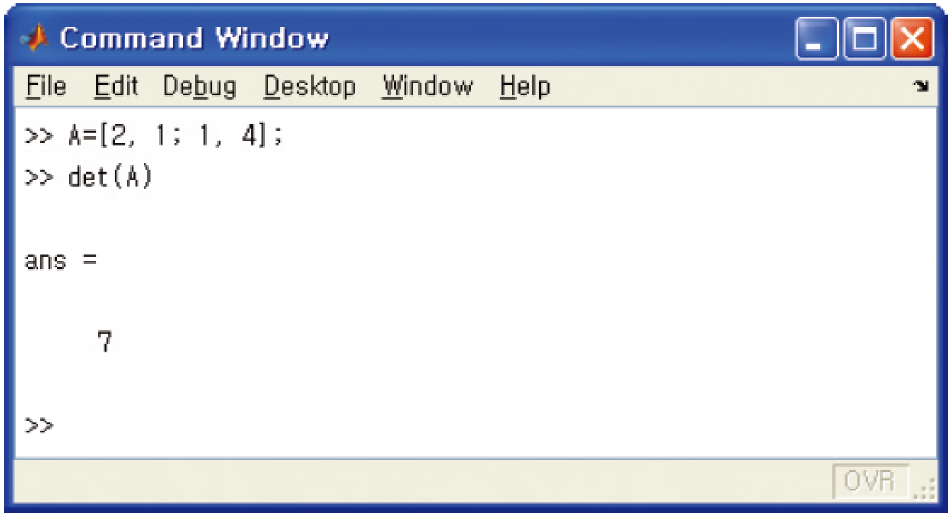
실습 ❸-8

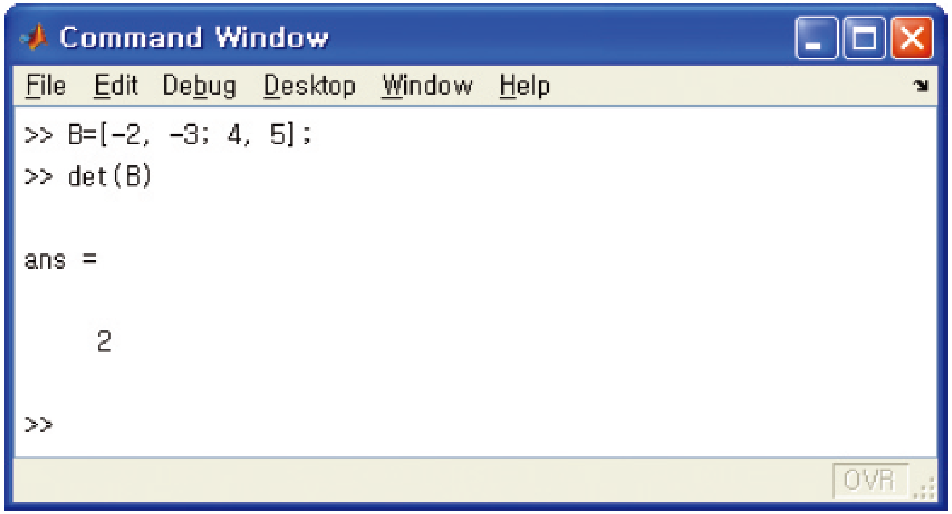
다음 행렬 A, B의 행렬식을 각각 구해 보자.

$$A=\begin{bmatrix}2 & 1\\ 1 & 4\end{bmatrix},\quad B=\begin{bmatrix}-2 & -3\\ 4 & 5\end{bmatrix}$$

풀이 A의 행렬식은 $2\cdot4-1\cdot1=7$이다.

또한 B의 행렬식은 $(-2)\cdot5-(-3)\cdot4=-10+12=2$이다. ■


Command Window
File Edit Debug Desktop Window Help
>> A=[2, 1; 1, 4];
>> det(A)
ans =
7
>>
OVR


Command Window
File Edit Debug Desktop Window Help
>> B=[-2, -3; 4, 5];
>> det(B)
ans =
2
>>
OVR

실습 ❸-9

예제 ③-3 C program, MATLAB

다음 행렬 A의 행렬식을 제1행에 대한 여인수들로 전개하여 구해 보자.

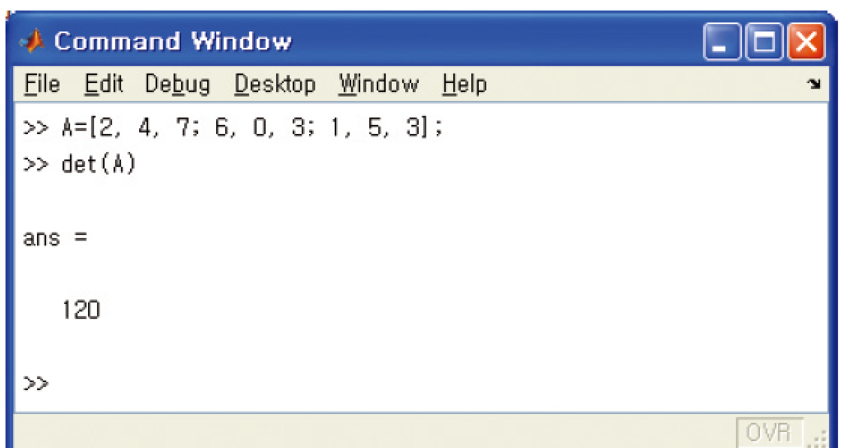
$$A=\begin{bmatrix}2 & 4 & 7\\ 6 & 0 & 3\\ 1 & 5 & 3\end{bmatrix}$$

풀이

$$\mathrm{Det}(A)=2(-1)^{1+1}\begin{vmatrix}0 & 3\\ 5 & 3\end{vmatrix}+4(-1)^{1+2}\begin{vmatrix}6 & 3\\ 1 & 3\end{vmatrix}+7(-1)^{1+3}\begin{vmatrix}6 & 0\\ 1 & 5\end{vmatrix}$$

$$=2(0\cdot 3-3\cdot 5)-4(6\cdot 3-3\cdot 1)+7(6\cdot 5-0\cdot 1)$$

$$=120 \quad ■$$



3.5 컴퓨터 프로그램에 의한 연산

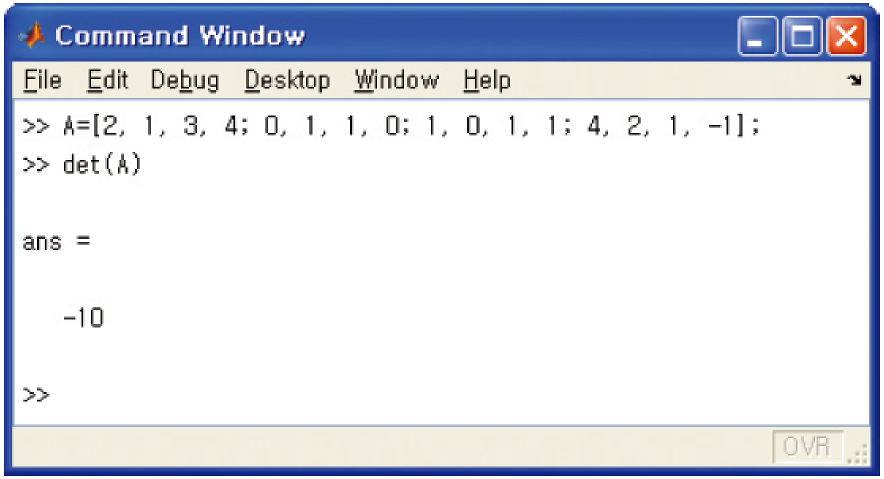
실습 ❸-10

예제 ③-9 | C program, MATLAB

행렬식의 성질을 이용하여 다음 행렬식의 값을 구해 보자.

$$|A| = \begin{vmatrix} 2 & 1 & 3 & 4 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \\ 4 & 2 & 1 & -1 \end{vmatrix}$$

풀이 $|A| = (1)(1)(-5)(2) = -10$ ■

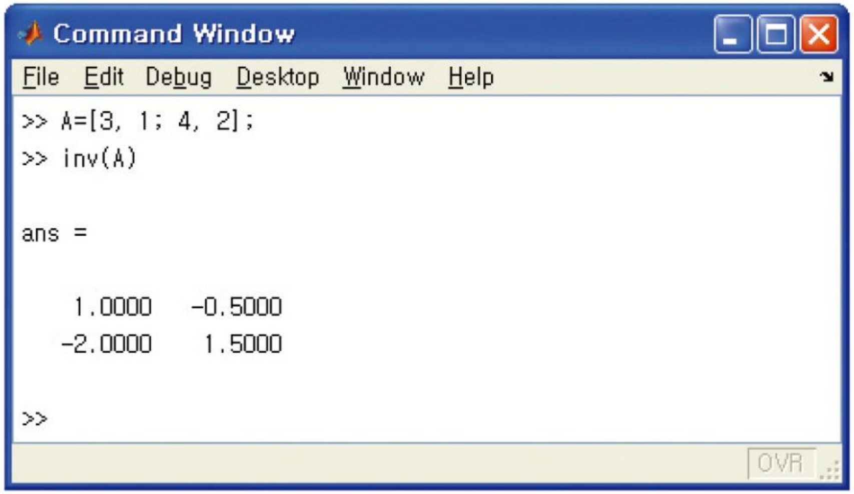


예제 3-20 C program, MATLAB

실습 3-11 $A=\begin{bmatrix} 3 & 1 \\ 4 & 2 \end{bmatrix}$일 때 수반행렬을 이용하여 A의 역행렬을 구해 보자.

풀이 $|A|=\begin{vmatrix} 3 & 1 \\ 4 & 2 \end{vmatrix}=2$이므로 가역적이다.

따라서 $A^{-1}=\frac{1}{2}\begin{bmatrix} 2 & -1 \\ -4 & 3 \end{bmatrix}$이 된다. ■



실습 ❸-12

예제 ③-21 | C program, MATLAB

다음에서 행렬식과 Adj(A)를 각각 구하고 역행렬이 존재하면 구해 보자.

$$(2)\ A=\begin{bmatrix} 3 & 1 & 0 \\ 2 & -1 & 1 \\ 5 & 5 & -7 \end{bmatrix}$$

풀이 먼저 행렬식을 구하고 Adj(A)를 만든 후 역행렬을 구한다.

$$(2)\ |A|=25,\quad \mathrm{Adj}(A)=\begin{bmatrix} 2 & 7 & 1 \\ 19 & -21 & -3 \\ 15 & -10 & -5 \end{bmatrix},$$

$$A^{-1}=\frac{1}{25}\begin{bmatrix} 2 & 7 & 1 \\ 19 & -21 & -3 \\ 15 & -10 & -5 \end{bmatrix}\ ■$$

컴퓨터 프로그램에 의한 연산

Command Window

File Edit Debug Desktop Window Help

```
>> A=[3, 1, 0; 2, -1, 1; 5, 5, -7];
>> inv(A)

ans =

    0.0800    0.2800    0.0400
    0.7600   -0.8400   -0.1200
    0.6000   -0.4000   -0.2000

>>
```

OVR

행렬식의 생활속의 응용

- 행렬식의 값이 0인지의 여부를 통해 역행렬을 구할 수 있는지를 즉각 판정할 수 있다. 따라서 제1장과 제2장에 기술된 응용 문제들을 빠르고 효율적으로 해결할 수 있다.
- 행렬식을 이용하여 특정한 암세포의 활동을 모델링하려는 연구가 시도되고 있다.
- 행렬식을 이용하여 좌표가 주어진 삼각형의 면적을 쉽게 구할 수 있다.
- 행렬식은 벡터미적분학에서 체적을 계산하는 데에 쓰일 수 있다. 실벡터들로 이루어진 행렬의 행렬식의 절대값은 그 벡터들을 각 변으로 가지는 평행육면체의 체적과 같다.
- 컴퓨터 보안이나 극비 문서의 보안을 위해 어떤 주어진 메시지를 암호화하는 방법으로는 정수론과 더불어 행렬이 많이 쓰인다. 암호로의 변환에는 행렬의 곱이 쓰이고, 전달받은 암호를 해독하기 위해서는 역행렬이 활용된다.