

# 선형방정식의 해법과 응용

## 개요

- ❖ 선형방정식을 행렬을 이용한 첨가행렬로 바꾼 후
  - 행 사다리꼴 형태의 기본 행 연산을 함으로써 보다 편리하게 선형방정식의 해를 구하는 방법과
  - LU-분해법에 의한 선형방정식의 해법을 알아봄
- ❖ 선형방정식의 응용 부분에서는 선형방정식을 활용하여
  - 화학방정식, 교통 흐름, 마르코프 체인, 암호 해독, 키르히호프의 법칙 등 다양한 응용들을 살펴봄

# 선형방정식의 해법과 응용

## CONTENTS

### 4.1 가우스 소거법을 이용한 선형방정식의 해법

4.1.1 첨가행렬로의 표현

4.1.2 가우스-조단 소거법에 의한 선형방정식의 해법

4.1.3 LU-분해법에 의한 선형방정식의 해법

### 4.2 선형방정식의 다양한 응용들

4.2.1 여러 가지 응용들

4.2.2 화학방정식에의 응용

4.2.3 교통 흐름에의 응용

4.2.4 마르코프 체인에의 응용

4.2.5 암호 해독에의 응용

4.2.6 키르히호프 법칙에의 응용

### 4.3 C Program에 의한 선형방정식의 해법(가우스-조단 소거법)

## 4.1 가우스 소거법을 이용한 선형방정식의 해법



### 예제 ④-1

다음과 같은 선형시스템에서 계수행렬과 첨가행렬을 구해 보자.

$$\begin{aligned} 2x_1 + 3x_2 - 4x_3 + x_4 &= 5 \\ -2x_1 &\quad + x_3 = 7 \\ 3x_1 + 2x_2 &\quad - 4x_4 = 3 \end{aligned}$$

**풀이** 위의 식을 행렬의 곱 형태로 나타내면 다음과 같다.

$$\begin{bmatrix} 2 & 3 & -4 & 1 \\ -2 & 0 & 1 & 0 \\ 3 & 2 & 0 & -4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} 5 \\ 7 \\ 3 \end{bmatrix}$$

그러면 이 시스템의 계수행렬은

## 4.1 가우스 소거법을 이용한 선형방정식의 해법

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

이고, 첨가행렬은 다음과 같다.

$$\left[ \begin{array}{cccc|c} 2 & 3 & -4 & 1 & 5 \\ -2 & 0 & 1 & 0 & 7 \\ 3 & 2 & 0 & -4 & 3 \end{array} \right] \blacksquare$$

# 4.1 가우스 소거법을 이용한 선형방정식의 해법



## 예제 4-2

다음 선형시스템을  $Ax = b$ 의 형태로 나타내고, 계수행렬과 첨가행렬을 구해 보자.

$$x + 2y - 3z = -3$$

$$3x + 3y - 4z = 2$$

$$4x - 2y + 6z = -1$$

**풀이** 이 선형시스템의 계수행렬  $A$ 와 첨가행렬  $[A|b]$ 는 다음과 같이 구할 수 있다.

$$A = \begin{bmatrix} 1 & 2 & -3 \\ 3 & 3 & -4 \\ 4 & -2 & 6 \end{bmatrix}, \quad x = \begin{bmatrix} x \\ y \\ z \end{bmatrix}, \quad b = \begin{bmatrix} -3 \\ 2 \\ -1 \end{bmatrix} \text{ 일 때 } Ax = b \text{ 가 된다.}$$

또한 첨가행렬은

$$[A|b] = \left[ \begin{array}{ccc|c} 1 & 2 & -3 & -3 \\ 3 & 3 & -4 & 2 \\ 4 & -2 & 6 & -1 \end{array} \right] \text{이 된다. } \blacksquare$$

## 4.1 가우스 소거법을 이용한 선형방정식의 해법



### 정리 ④-1

다음과 같은 선형시스템에서의 소거법과 첨가행렬에서의 기본 행 연산은 동치이다.

#### (1) 선형시스템에서의 가우스 소거법

- ① 한 방정식에 0이 아닌 상수  $k$ 를 곱한다.
- ② 선형시스템의 두 방정식을 바꾼다.
- ③ 한 방정식을  $k$  ( $\neq 0$ )배하여 다른 방정식에 더한다.

#### (2) 첨가행렬에서의 기본 행 연산

- ① 한 행에 0이 아닌 상수  $k$ 를 곱한다.
- ② 두 행을 바꾼다.
- ③ 한 행을  $k$  ( $\neq 0$ )배하여 다른 행에 더한다.

## 4.1 가우스 소거법을 이용한 선형방정식의 해법



여기서 잠깐!!

여기서 다루는 가우스-조단 소거법은 행렬의 행 간의 연산이다. 이러한 행 연산을 통하여 행렬로 구성된 방정식의 해를 구할 수 있다. 또한 가우스-조단 방법으로 행 연산(row operation)을 하는 것을 행 줄임(row reduce)이라고도 한다. 선형시스템에서 변수를 가지고 소거해 나가는 것보다는 기본 행 연산으로 하는 것이 결과는 같으면서도 훨씬 편리하다.

## 4.1 가우스 소거법을 이용한 선형방정식의 해법



### 예제 4-3

C program

다음 선형시스템의 해를 가우스-조단 소거법을 이용하여 구해 보자.

$$-x_2 - x_3 + x_4 = 0$$

$$x_1 + x_2 + x_3 + x_4 = 6$$

$$2x_1 + 4x_2 + x_3 - 2x_4 = -1$$

$$3x_1 + x_2 - 2x_3 + 2x_4 = 3$$

**풀이** 주어진 시스템의 첨가행렬은 다음과 같다.

$$\left[ \begin{array}{cccc|c} 0 & -1 & -1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 6 \\ 2 & 4 & 1 & -2 & -1 \\ 3 & 1 & -2 & 2 & 3 \end{array} \right]$$

여기서 0을 피벗으로 사용할 수 없으므로 첫째 행과 둘째 행을 서로 바꾼다. 따라서 새로운 첫 번째 행이 피벗 행이 되고 피벗은 1이 된다.

## 4.1 가우스 소거법을 이용한 선형방정식의 해법

피벗  $\rightarrow$

$$\left[ \begin{array}{cccc|c} 1 & 1 & 1 & 1 & 6 \\ 0 & -1 & -1 & 1 & 0 \\ 2 & 4 & 1 & -2 & -1 \\ 3 & 1 & -2 & 2 & 3 \end{array} \right]$$

피벗행

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

$$(-3) \times R_1 + R_4 \rightarrow R_4$$

$$\left[ \begin{array}{cccc|c} 1 & 1 & 1 & 1 & 6 \\ 0 & -1 & -1 & 1 & 0 \\ 0 & 2 & -1 & -4 & -13 \\ 0 & -2 & -5 & -1 & -15 \end{array} \right]$$

$$2 \times R_2 + R_3 \rightarrow R_3$$

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

$$\left[ \begin{array}{cccc|c} 1 & 1 & 1 & 1 & 6 \\ 0 & -1 & -1 & 1 & 0 \\ 0 & 0 & -3 & -2 & -13 \\ 0 & 0 & -3 & -3 & -15 \end{array} \right]$$

$$(-1) \times R_3 + R_4 \rightarrow R_4$$

## 4.1 가우스 소거법을 이용한 선형방정식의 해법

$$\left[ \begin{array}{cccc|c} 1 & 1 & 1 & 1 & 6 \\ 0 & -1 & -1 & 1 & 0 \\ 0 & 0 & -3 & -2 & -13 \\ 0 & 0 & 0 & -1 & -2 \end{array} \right]$$

그 결과 이 첨가행렬은 행 사다리꼴이 되고 역대입법에 의해

$x_1 = 2, x_2 = -1, x_3 = 3, x_4 = 2$ 인 해를 구할 수 있다. ■

## 4.1 가우스 소거법을 이용한 선형방정식의 해법



### 예제 ④-4

C program

다음의 선형시스템이 해를 가지는지를 판별해 보자.

$$x_1 - 2x_2 + x_3 = 0$$

$$2x_2 - 8x_3 = 8$$

$$-4x_1 + 5x_2 + 9x_3 = -9$$

**풀이** 삼각형 형태(triangular form)를 얻기 위해 필요한 행 연산을 하면 다음과 같다.

$$x_1 - 2x_2 + x_3 = 0$$

$$x_2 - 4x_3 = 4$$

$$x_3 = 3$$

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

이 단계에서 세 번째 식으로부터  $x_3 = 3$ 임을 알 수 있다. 그리고는  $x_3$ 의 값을 두 번째 식에 대입하여  $x_2 = 16$ 의 값을 구하고, 이 값을 첫 번째 식에 대입하여  $x_1 = 29$ 의 값을 구할 수 있다. 따라서 이 시스템은  $x_1 = 29$ ,  $x_2 = 16$ ,  $x_3 = 3$ 인 유일한 해를 가진다. ■

## 4.1 가우스 소거법을 이용한 선형방정식의 해법



### 예제 4-5

다음의 선형시스템이 해를 가지는지를 판별해 보자.

$$x_2 - 4x_3 = 8$$

$$2x_1 - 3x_2 + 2x_3 = 1$$

$$5x_1 - 8x_2 + 7x_3 = 1$$

**풀이** 이 시스템을 첨가행렬로 나타내면 다음과 같다.

$$\left[ \begin{array}{ccc|c} 0 & 1 & -4 & 8 \\ 2 & -3 & 2 & 1 \\ 5 & -8 & 7 & 1 \end{array} \right]$$

첫 번째 행의 첫 항이 0이므로 첫 번째 행과 두 번째 행을 서로 교환한다.

$$\left[ \begin{array}{ccc|c} 2 & -3 & 2 & 1 \\ 0 & 1 & -4 & 8 \\ 5 & -8 & 7 & 1 \end{array} \right]$$

$$\left( -\frac{5}{2} \right) \times R_1 + R_3 \rightarrow R_3$$

## 4.1 가우스 소거법을 이용한 선형방정식의 해법

$$\left[ \begin{array}{ccc|c} 2 & -3 & 2 & 1 \\ 0 & 1 & -4 & 8 \\ 0 & -\frac{1}{2} & 2 & -\frac{3}{2} \end{array} \right]$$

$$\frac{1}{2} \times R_2 + R_3 \rightarrow R_3$$

$$\left[ \begin{array}{ccc|c} 2 & -3 & 2 & 1 \\ 0 & 1 & -4 & 8 \\ 0 & 0 & 0 & \frac{5}{2} \end{array} \right]$$

그 결과 첨가행렬이 상부삼각행렬로 바뀌었다. 이것을 원래의 방정식 형태로 나타내면 다음과 같은 선형시스템을 얻는다.

$$2x_1 - 3x_2 + 2x_3 = 1$$

$$x_2 - 4x_3 = 8$$

$$0 = \frac{5}{2}$$

$0 = \frac{5}{2}$ 는 원래의 방정식  $0x_1 + 0x_2 + 0x_3 = \frac{5}{2}$ 로 명백한 모순이다. 따라서 주어진 식을 만족시키는 해  $x_1, x_2, x_3$ 이 존재하지 않는다. ■

## 4.1 가우스 소거법을 이용한 선형방정식의 해법



### 예제 4-6

다음의 동차 선형시스템에서 해의 개수를 판별해 보자.

$$x_1 + x_2 + x_3 + x_4 = 0$$

$$x_1 + x_4 = 0$$

$$x_1 + 2x_2 + x_3 = 0$$

**풀이** 먼저 이 시스템의 첨가행렬을 구하면

$$A = \left[ \begin{array}{cccc|c} 1 & 1 & 1 & 1 & | & 0 \\ 1 & 0 & 0 & 1 & | & 0 \\ 1 & 2 & 1 & 0 & | & 0 \end{array} \right]$$

가 된다.

## 4.1 가우스 소거법을 이용한 선형방정식의 해법

이것에다 행 연산을 반복하면

$$\left[ \begin{array}{cccc|c} 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & -1 & 0 \\ 0 & 0 & 1 & 1 & 0 \end{array} \right]$$

과 같은 행 사다리꼴이 만들어진다. 따라서 구하는 해는 다음과 같다.

$$x_1 = -r$$

$$x_2 = r$$

$$x_3 = -r$$

$$x_4 = r \quad (r \text{은 임의의 실수})$$

그러므로 이 선형시스템의 해의 개수는 무한히 많다. ■

# 4.1 가우스 소거법을 이용한 선형방정식의 해법



## 예제 4-7

다음과 같이 계수가 같은 두 개의 선형시스템에서 두 식의 해를 동시에 구해 보자.

$$(1) \quad x_1 + 2x_2 = 2$$

$$3x_1 + 7x_2 = 8$$

$$(2) \quad x_1 + 2x_2 = 1$$

$$3x_1 + 7x_2 = 7$$

**풀이** 2개의 선형시스템은 크기도 같고 같은 계수행렬을 가지므로, 계수행렬의 오른쪽에 상수항의 열들을 첨가하면 다음과 같은 첨가행렬을 만들 수 있다.

$$\left[ \begin{array}{cc|cc} 1 & 2 & 2 & 1 \\ 3 & 7 & 8 & 7 \end{array} \right]$$

$$(-3) \times R_1 + R_2 \rightarrow R_2$$

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

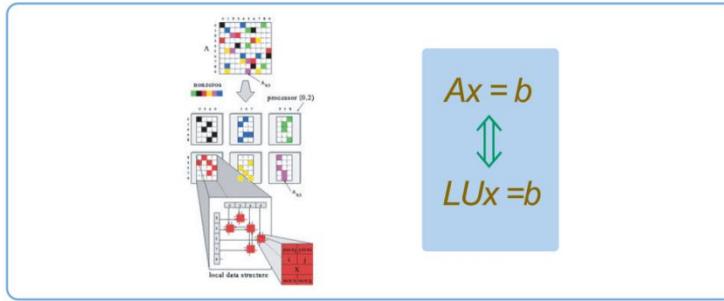
$$(-2) \times R_2 + R_1 \rightarrow R_1$$

$$\left[ \begin{array}{cc|cc} 1 & 0 & -2 & -7 \\ 0 & 1 & 2 & 4 \end{array} \right]$$

따라서 (1)식의 해는  $x_1 = -2, x_2 = 2$ 이고 (2)식의 해는  $x_1 = -7, x_2 = 4$ 이다. ■

## 4.1 가우스 소거법을 이용한 선형방정식의 해법

### 4.1.3 LU-분해법에 의한 선형방정식의 해법



선형시스템의 해를 구하는 또 다른 방법으로는 먼저 정방행렬을 하부삼각행렬  $L$ 과 상부삼각행렬  $U$ 의 곱의 형태로 인수분해하여 구하는 방법이다. 즉,  $Ax = b$ 에서  $A$ 를  $A = LU$ 로 인수분해하여 중간 해를 구한 후 다시 원래의 해를 구하게 되는데, 이러한 방법을 **LU-분해법(LU-decomposition)**이라고 한다.  $L$ 과  $U$ 가 둘 다 삼각행렬이기 때문에 인수분해하는 과정을 제외하고는 일반적으로 계산이 쉬운 편이다.  $L$ 과  $U$ 는 원래의 행렬을 가우스 소거법을 이용하여 하부삼각행렬과 상부삼각행렬로 각각 만들 수 있다.

$LU$ 를 구하는 알고리즘

- (1) 주어진 행렬  $A$ 에 가우스 소거법을 적용하여 상부삼각행렬  $U$ 를 구한다.
- (2) 단위행렬  $I$ 에 (1) 연산의 역( $+ \leftrightarrow -, \times \leftrightarrow \div$ )을 역순으로 적용하여  $L$ 을 구한다.

# 4.1 가우스 소거법을 이용한 선형방정식의 해법



## 예제 4-8

다음의 행렬  $A$ 로부터  $U$ 와  $L$ 을 구해 보자.

$$A = \begin{bmatrix} 1 & 2 & 1 \\ 2 & 3 & 3 \\ -3 & -10 & 2 \end{bmatrix}$$

### 풀이

$$A = \begin{bmatrix} 1 & 2 & 1 \\ 2 & 3 & 3 \\ -3 & -10 & 2 \end{bmatrix} \quad (-2)R_1 + R_2 \rightarrow R_2$$

$$\begin{bmatrix} 1 & 2 & 1 \\ 0 & -1 & 1 \\ -3 & -10 & 2 \end{bmatrix} \quad 3R_1 + R_3 \rightarrow R_3$$

$$\begin{bmatrix} 1 & 2 & 1 \\ 0 & -1 & 1 \\ 0 & -4 & 5 \end{bmatrix} \quad (-4)R_2 + R_3 \rightarrow R_3$$

## 4.1 가우스 소거법을 이용한 선형방정식의 해법

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

$L$ 을 구하기 위해  $I$ 로부터 시작한다.

$$I = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$4R_2 + R_3 \rightarrow R_3$$

$$I = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 4 & 1 \end{bmatrix}$$

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

$$I = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -3 & 4 & 1 \end{bmatrix}$$

$$2R_1 + R_2 \rightarrow R_2$$

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

$$= L \blacksquare$$



## 알고리즘

*LU*-분해법(*LU*-decomposition)은 다음의 4단계로 이루어진다.

**1단계** 선형시스템  $Ax = b$ 를  $LUX = b$ 로 분해한다.

**2단계** 새로운 변수  $y$ 를  $Ux = y$ 로 나타내고 원래의 시스템을  $Ly = b$ 로 대체한다.

**3단계**  $Ly = b$ 를 변수 벡터  $y$ 에 대하여 해를 구한다.

**4단계** 벡터  $y$ 를  $Ux = y$ 에 대입하여  $x$ 에 대하여 최종 해를 구한다.

## 4.1 가우스 소거법을 이용한 선형방정식의 해법



### 예제 4-9

다음과 같이 주어진 선형시스템에서  $LU$ -분해법에 의해 해를 구해 보자.

$$x_1 + x_2 + x_3 = 7$$

$$x_1 + 2x_2 + 2x_3 = 7$$

$$x_1 + 2x_2 + 3x_3 = 5$$

**풀이** 주어진 식을 다음과 같이  $Ax = b$ 의 형태로 바꾼 후,  $A = LU$ 로  
인수분해되었다고 가정한다.

$$\begin{bmatrix} 1 & 1 & 1 \\ 1 & 2 & 2 \\ 1 & 2 & 3 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 7 \\ 7 \\ 5 \end{bmatrix}$$

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

$L$                    $U$

## 4.1 가우스 소거법을 이용한 선형방정식의 해법

$Ax = b$ 를 다음과 같이  $LUX = b$ 와 같이 표현한다.

$$\begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 7 \\ 7 \\ 5 \end{bmatrix}$$

$L \qquad \qquad U \qquad x = b$

**2단계**  $Ux = y$ 로 나타내고  $y$ 를  $y_1, y_2, y_3$ 의 열벡터라고 하면 원래의 식은 다음과 같다.

$$\begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} 7 \\ 7 \\ 5 \end{bmatrix}$$

$L \qquad y = b$

## 4.1 가우스 소거법을 이용한 선형방정식의 해법

3단계  $Ly = b$ 를 변수 벡터  $y$ 에 대하여 해를 구한다.

$$y_1 = 7$$

$$y_1 + y_2 = 7$$

$$y_1 + y_2 + y_3 = 5$$

$y$ 의 벡터 값을 구하면

$$y_1 = 7, \quad y_2 = 0, \quad y_3 = -2 \text{이다.}$$

4단계 벡터  $y$ 를  $Ux = y$ 에 대입하여  $x$ 에 대한 해를 구한다.

$$\begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 7 \\ 0 \\ -2 \end{bmatrix}$$

## 4.1 가우스 소거법을 이용한 선형방정식의 해법

따라서

$$x_1 + x_2 + x_3 = 7$$

$$x_2 + x_3 = 0$$

$$x_3 = -2$$

이 식에다 역대입법을 적용하면 주어진 선형시스템의 최종 해인  $x_1=7$ ,  $x_2=2$ ,  $x_3=-2$ 를 얻는다. ■

## 4.1 가우스 소거법을 이용한 선형방정식의 해법



### 예제 4-10

다음과 같이 주어진 시스템에서  $LU$ -분해법에 의해 해를 구해 보자.

$$\begin{array}{rcl} 6x_1 - 2x_2 - 4x_3 + 4x_4 & = & 2 \\ 3x_1 - 3x_2 - 6x_3 + x_4 & = & -4 \\ -12x_1 + 8x_2 + 21x_3 - 8x_4 & = & 8 \\ -6x_1 - 10x_3 + 7x_4 & = & -43 \end{array}$$

#### 풀이

1단계 주어진 식을 다음과 같이  $Ax = b$ 의 형태로 바꾼 후,  $A = LU$ 로  
인수분해되었다고 가정한다.

$$\left[ \begin{array}{rrrr|c} 6 & -2 & -4 & 4 & x_1 \\ 3 & -3 & -6 & 1 & x_2 \\ -12 & 8 & 21 & -8 & x_3 \\ -6 & 0 & -10 & 7 & x_4 \end{array} \right] = \left[ \begin{array}{c} 2 \\ -4 \\ 8 \\ -43 \end{array} \right]$$

## 4.1 가우스 소거법을 이용한 선형방정식의 해법

$$A = \begin{bmatrix} 6 & -2 & -4 & 4 \\ 3 & -3 & -6 & 1 \\ -12 & 8 & 21 & -8 \\ -6 & 0 & -10 & 7 \end{bmatrix} = \begin{bmatrix} 2 & 0 & 0 & 0 \\ 1 & -1 & 0 & 0 \\ -4 & 2 & 1 & 0 \\ -2 & -1 & -2 & 2 \end{bmatrix} \begin{bmatrix} 3 & -1 & -2 & 2 \\ 0 & 2 & 4 & 1 \\ 0 & 0 & 5 & -2 \\ 0 & 0 & 0 & 4 \end{bmatrix}$$

*L*                            *U*

$Ax = b$ 를 다음과 같이  $LUX = b$ 와 같이 표현한다.

$$\begin{bmatrix} 2 & 0 & 0 & 0 \\ 1 & -1 & 0 & 0 \\ -4 & 2 & 1 & 0 \\ -2 & -1 & -2 & 2 \end{bmatrix} \begin{bmatrix} 3 & -1 & -2 & 2 \\ 0 & 2 & 4 & 1 \\ 0 & 0 & 5 & -2 \\ 0 & 0 & 0 & 4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} 2 \\ -4 \\ 8 \\ -43 \end{bmatrix}$$

*L*                            *U*                             $x = b$

## 4.1 가우스 소거법을 이용한 선형방정식의 해법

2단계  $Ux = y$ 로 나타내고  $y$ 를  $y_1, y_2, y_3, y_4$ 의 열벡터라고 하면 원래의 식은 다음과 같다.

$$\left[ \begin{array}{cccc} 2 & 0 & 0 & 0 \\ 1 & -1 & 0 & 0 \\ -4 & 2 & 1 & 0 \\ -2 & -1 & -2 & 2 \end{array} \right] \left[ \begin{array}{c} y_1 \\ y_2 \\ y_3 \\ y_4 \end{array} \right] = \left[ \begin{array}{c} 2 \\ -4 \\ 8 \\ -43 \end{array} \right]$$

3단계  $Ly = b$ 를 변수 벡터  $y$ 에 대하여 해를 구한다.

$$y_1 = 1$$

$$y_2 = \frac{-4 - y_1}{-1} = 5$$

$$y_3 = 8 + 4y_1 - 2y_2 = 2$$

$$y_4 = \frac{-43 + 2y_1 + y_2 + 2y_3}{2} = -16$$

## 4.1 가우스 소거법을 이용한 선형방정식의 해법

4단계 벡터  $y$ 를  $Ux = y$ 에 대입하여  $x$ 에 대한 해를 구한다.

$$\left[ \begin{array}{cccc|c} 3 & -1 & -2 & 2 \\ 0 & 2 & 4 & 1 \\ 0 & 0 & 5 & -2 \\ 0 & 0 & 0 & 4 \end{array} \right] \begin{matrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{matrix} = \begin{bmatrix} 1 \\ 5 \\ 2 \\ -16 \end{bmatrix}$$

따라서

$$x_4 = -4$$

$$x_3 = \frac{2 + 2x_4}{5} = -1.2$$

$$x_2 = \frac{5 - 4x_3 - x_4}{2} = 6.9$$

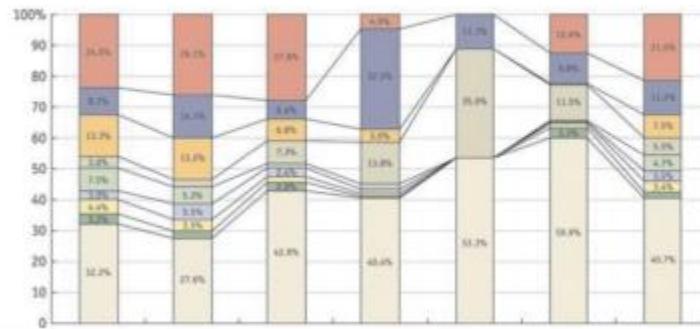
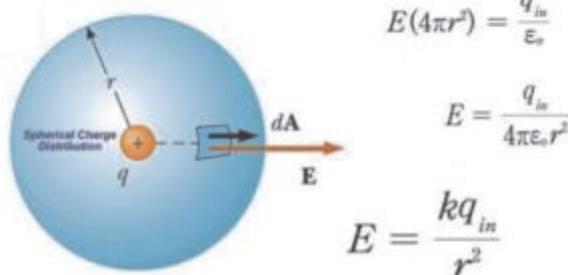
$$x_1 = \frac{1 + x_2 + 2x_3 - 2x_4}{3} = 4.5$$

를 얻는다. ■

## 4.2.1 여러 가지 응용들

Gauss's Law to Coulomb's Law

$$\Phi_E = \oint E \cdot dA = \int E dA = E \int dA = \frac{q_{in}}{\epsilon_0}$$

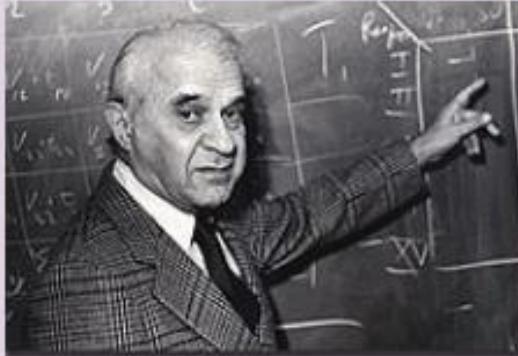
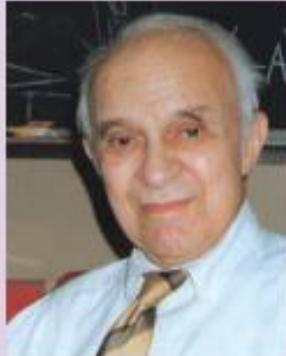




## 예제 4-11

경제학에서 종종 나타나는 모형 중의 하나로 레온티에프(Leontief)의 입출력 모형이 있다.  $n$ 개의 산업체가 있고 각 산업체는 두 종류의 수요가 있다고 하자. 첫 번째는 그 시스템의 외부에서 오는 외적 수요이다. 가령 시스템이 한 주(state)에 속하면, 외적 수요는 다른 주에서 오는 것이다. 두 번째는 같은 시스템 내의 한 업체에서 다른 업체로의 수요이다. 가령 미국 내에서 강철업계의 생산품이 자동차업계에 소요되는 것과 비슷한 수요이다. 이러한 문제들은 모두 선형방정식으로 풀 수 있다.

레온티에프의 모델은 경제학에서는 고전적인 모델로 여겨지지만 선형방정식을 이용한 자세한 풀이를 하기에는 너무나 변수가 많아지므로 여기서는 개념만 간단히 소개하였다.



레온티에프(Wassily Leontief, 1906~1999)

진 선형방정식을 56시간에 걸쳐 풀었다. 레온티에프는 통계학에 사용되는 수학적 공식들의 경제적 의미, 응용 및 오용에 관한 연구에 매진하였으며 그 결과가 바로 투입-산출 분석이다. 그의 투입-산출 분석은 경제 계획 및 예측을 위해 각국에서 다양한 형태로 사용되고 있으며, 이러한 투입-산출 분석의 개발과 그 응용에 대한 업적으로 1973년 노벨 경제학상을 수상하였다.

러시아 출신의 경제학자인 레온티에프는 1925년 독일로 떠나 공부를 마친 후 1931년 미국으로 이주하여 ‘투입-산출 모델’의 발전과 응용에 주력하였다. 그는 1949년 MARK II 컴퓨터에서 42개의 변수를 가



## 예제 ④-12

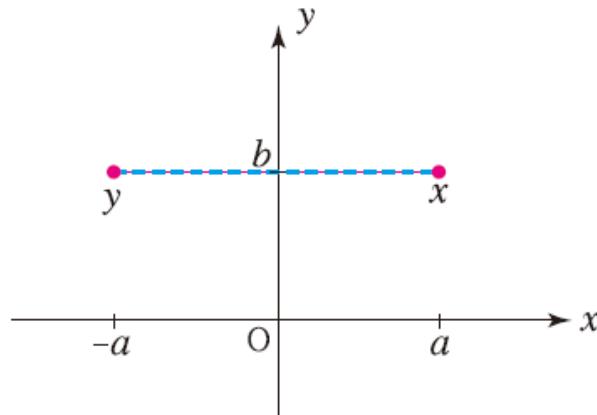
다음과 같이 행렬  $A$ ,  $x$ 가 주어졌을 때 행렬의 곱을 통한 함수에의 응용 예를 살펴보자.

$$A = \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix}, \quad x = \begin{bmatrix} a \\ b \end{bmatrix}$$

$A$ 와  $x$ 의 곱  $y = Ax$ 는

$$y = \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} -a \\ b \end{bmatrix}$$

이므로 행렬  $A$ 에 열벡터  $\begin{bmatrix} a \\ b \end{bmatrix}$ 를 곱한 값은 그 열벡터의 첫 번째 성분의 부호를 바꾼 것이 된다. 만약 열벡터가 평면의 점  $(a, b)$ 에 위치한 것으로 본다면 행렬  $A$ 를 곱한 결과는 〈그림 4.1〉과 같이 그 점을  $y$ 축에 대하여 반사한 것이 된다. ■



〈그림 4.1〉 한 점의  $y$ 축에 대한 반사



## 예제 4-13

다음과 같은 행렬의 곱을 통한 선형변환을 살펴보자.

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos \theta & \sin \theta \\ -\sin \theta & \cos \theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

이 선형변환은  $\theta = 60^\circ$ 일 때,  $\begin{bmatrix} 1 \\ 1 \end{bmatrix}, \begin{bmatrix} 1 \\ 2 \end{bmatrix}, \begin{bmatrix} 2 \\ 2 \end{bmatrix}, \begin{bmatrix} 2 \\ 1 \end{bmatrix}$ 과 같은 행렬들의 열벡터들을 곱함으로써  $60^\circ$ 의 각도로 회전시키는 역할을 한다.

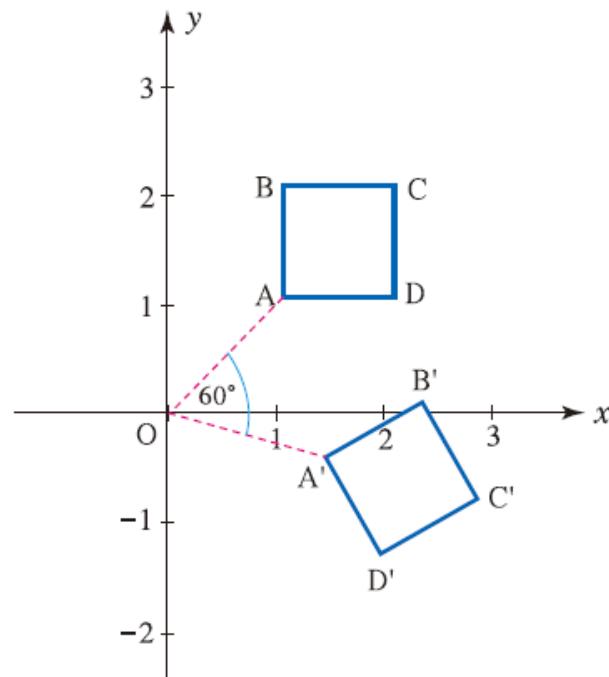
**풀이** 주어진 벡터들을 차례로 다음의 식에  $\begin{bmatrix} x \\ y \end{bmatrix}$ 로 대입하면

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 0.5 & 0.8660 \\ -0.8660 & 0.5 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

다음과 같은  $\begin{bmatrix} x' \\ y' \end{bmatrix}$  벡터들을 얻는다.(참고 :  $\cos 60^\circ = 0.5$ ,  $\sin 60^\circ = 0.8660$ )

$$\begin{bmatrix} 1.366 \\ -0.366 \end{bmatrix}, \begin{bmatrix} 2.232 \\ 0.134 \end{bmatrix}, \begin{bmatrix} 2.732 \\ -0.732 \end{bmatrix}, \begin{bmatrix} 1.866 \\ -1.232 \end{bmatrix}$$

이 점들을 <그림 4.2>와 같이 평면 위에 그리면 정사각형이 원래의 지점으로부터 원점을 축으로  $60^\circ$  회전한 것을 알 수 있다. 이러한 형태의 분석은 컴퓨터 스크린 상의 그림을 선형변환하는 기초를 형성하며 CAD/CAM에 매우 유용하다. ■



〈그림 4.2〉 선형변환된 정사각형



여기서 잠깐!!

여기서는 점의 대칭과 회전에 대해서만 간단한 예로 들었다. 선형변환과 관련된 더 다양하고 자세한 응용은 제9장의 선형변환에서 상세히 설명하였다. 선형변환은 이 외에도 주어진 벡터에다 적절한 행렬을 곱함으로써 확대, 축소 등의 연산을 매우 편리하게 할 수 있는 등 여러 가지 응용에 널리 활용되고 있다.



## 예제 4-14

선형방정식을 이용하여 평면 위의 두 점  $P(3, 4)$ 와  $Q(5, 6)$ 를 지나는 직선의 방정식을 구해 보자.

**풀이** 구하려는 직선의 방정식을  $ax + by + c = 0$ 이라고 하면 점  $P(3, 4)$ 와  $Q(5, 6)$ 가 직선 위에 있으므로  $3a + 4b + c = 0$ 과  $5a + 6b + c = 0$ 이 성립한다.

그러므로  $3a + 4b + c = 0$ ,  $5a + 6b + c = 0$ ,  $ax + by + c = 0$ 을 만족하면서 동시에 모두 0이 되지 않는  $a$ ,  $b$ ,  $c$ 를 구하면 된다. 즉,  $a$ ,  $b$ ,  $c$ 를 변수로 하는 다음과 같은 선형시스템이 비자명해를 가지게 되는 경우이다.

$$\begin{bmatrix} x & y & 1 \\ 3 & 4 & 1 \\ 5 & 6 & 1 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

따라서

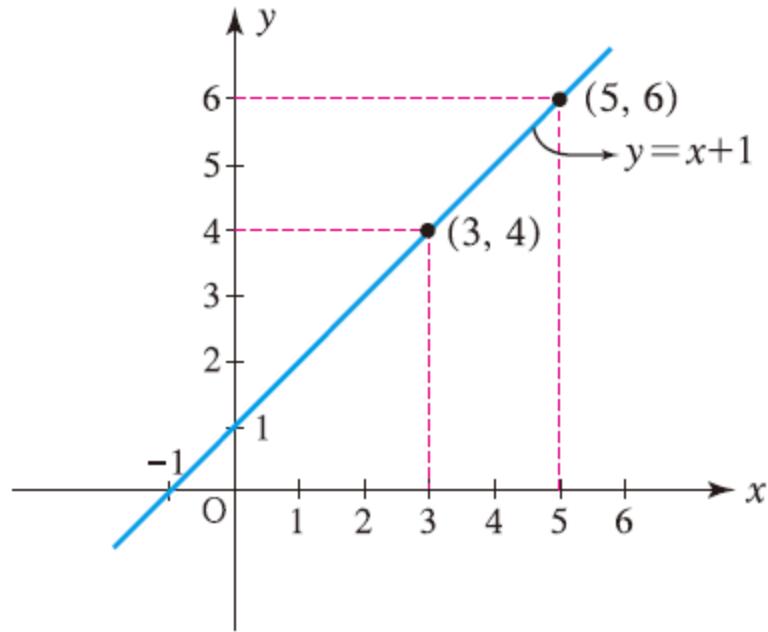
$$\begin{bmatrix} x & y & 1 \\ 3 & 4 & 1 \\ 5 & 6 & 1 \end{bmatrix} = 0$$

1행에 대하여 여인수들로 전개하면

$$x \begin{vmatrix} 4 & 1 \\ 6 & 1 \end{vmatrix} - y \begin{vmatrix} 3 & 1 \\ 5 & 1 \end{vmatrix} + 1 \begin{vmatrix} 3 & 4 \\ 5 & 6 \end{vmatrix} = 0$$

$$-2x + 2y - 2 = 0$$

따라서  $y = x + 1$ 과 같은 직선 방정식을 구할 수 있다. ■



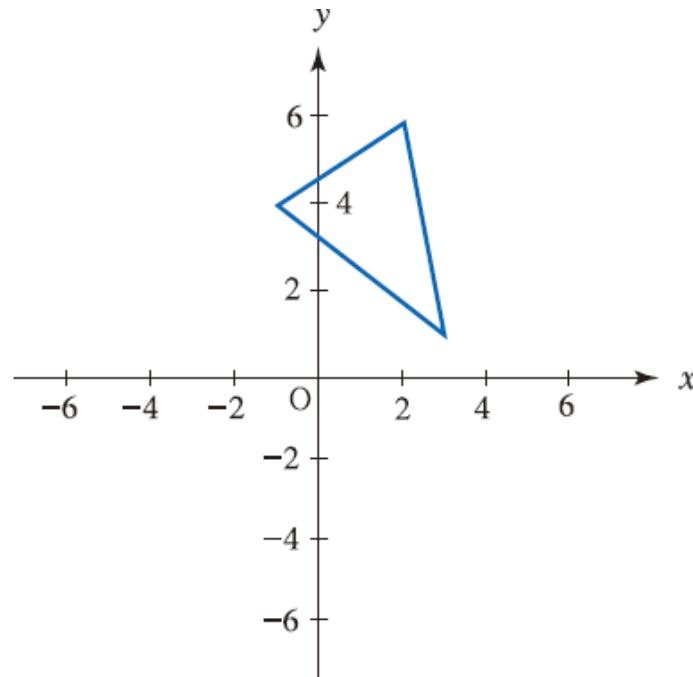
〈그림 4.3〉 두 점을 지나는 직선 방정식



## 예제 4-15

다음과 같이 세 개의 꼭지점이  $(x_1, y_1), (x_2, y_2), (x_3, y_3)$ 인 삼각형의 면적은

$\frac{1}{2} \times \begin{vmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ x_3 & y_3 & 1 \end{vmatrix}$ 이다. 이 경우 다음 <그림 4.4>의 삼각형의 면적을 구해 보자.



<그림 4.4> 행렬식을 이용한 삼각형의 면적 구하기

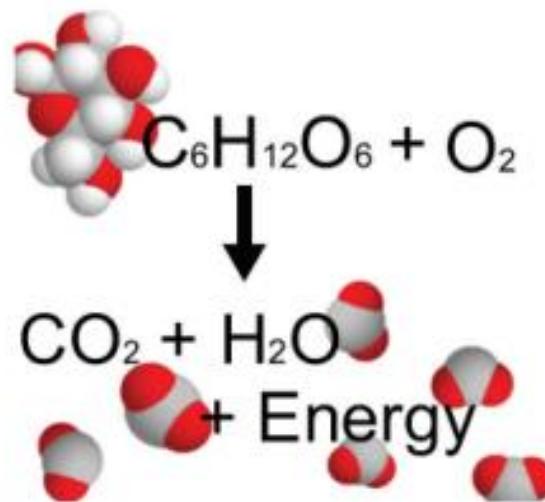
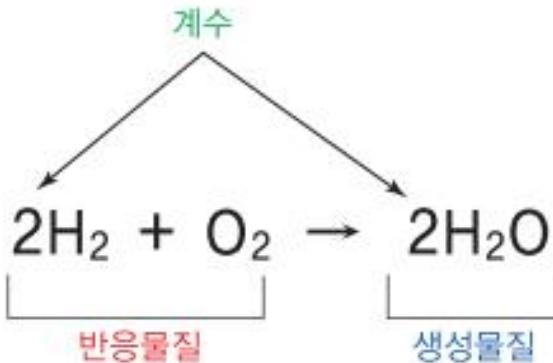
**풀이** 주어진 삼각형의 면적은

$$\frac{1}{2} \times \begin{vmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ x_3 & y_3 & 1 \end{vmatrix} \text{이므로}$$

이 식에다 좌표 값들을 적용하여 그 값을 구하면 삼각형의 면적을 비교적 쉽게 구할 수 있다. 즉,

$$\frac{1}{2} \times \begin{vmatrix} -1 & 4 & 1 \\ 3 & 1 & 1 \\ 2 & 6 & 1 \end{vmatrix} = \frac{1}{2} \times |17| = 8.5 \text{가 된다. } \blacksquare$$

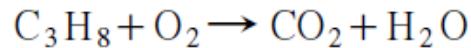
#### 4.2.2 화학방정식에의 응용



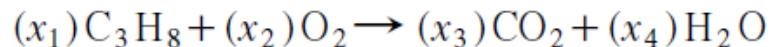


## 예제 4-16

다음 화학방정식의 균형(Balancing Chemical Equations)을 맞추어 보자.



**풀이** 화학방정식은 화학 반응에 의해 소비되고 생산되는 물질의 양에 대한 방정식을 나타낸다. 예를 들어, 프로판가스가 연소될 때 다음의 방정식에 따라 프로판( $\text{C}_3\text{H}_8$ )은 공기 중의 산소( $\text{O}_2$ )와 결합하여 이산화탄소( $\text{CO}_2$ )와 물( $\text{H}_2\text{O}$ )을 생성하게 된다.



이 방정식의 균형을 맞추기 위해서는 탄소(C), 수소(H), 산소(O) 원자들의 총 개수인  $x_1, \dots, x_4$ 를 좌우가 균형이 맞도록 결정해야 한다.

화학방정식의 균형을 맞추는 체계적인 방법은 반응에서 나타난 각각의 원자들의 개수를 나타내는 벡터방정식을 만드는 것이다. 주어진 식은 세 가지 타입의 원자들을 포함하므로 각 분자당 원자의 개수를 나타내는 3차원의 벡터를 다음과 같이 만든다.

$$\text{C}_3\text{H}_8 : \begin{bmatrix} 3 \\ 8 \\ 0 \end{bmatrix}, \quad \text{O}_2 : \begin{bmatrix} 0 \\ 0 \\ 2 \end{bmatrix}, \quad \text{CO}_2 : \begin{bmatrix} 1 \\ 0 \\ 2 \end{bmatrix}, \quad \text{H}_2\text{O} : \begin{bmatrix} 0 \\ 2 \\ 1 \end{bmatrix} \quad \begin{array}{l} \leftarrow \text{탄소} \\ \leftarrow \text{수소} \\ \leftarrow \text{산소} \end{array}$$

이 식의 균형을 맞추기 위해  $x_1, \dots, x_4$  계수들은 다음 식을 만족시켜야 한다.

$$x_1 \begin{bmatrix} 3 \\ 8 \\ 0 \end{bmatrix} + x_2 \begin{bmatrix} 0 \\ 0 \\ 2 \end{bmatrix} = x_3 \begin{bmatrix} 1 \\ 0 \\ 2 \end{bmatrix} + x_4 \begin{bmatrix} 0 \\ 2 \\ 1 \end{bmatrix}$$

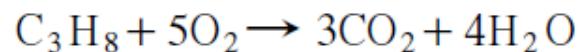
이것을 풀기 위해 모든 항들을 왼쪽으로 이항한다.

$$x_1 \begin{bmatrix} 3 \\ 8 \\ 0 \end{bmatrix} + x_2 \begin{bmatrix} 0 \\ 0 \\ 2 \end{bmatrix} + x_3 \begin{bmatrix} -1 \\ 0 \\ -2 \end{bmatrix} + x_4 \begin{bmatrix} 0 \\ -2 \\ -1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

다음의 식을 첨가행렬로 만들어 행 연산을 통해 다음과 같이 일반적인 해를 얻을 수 있다.

$$x_1 = \frac{1}{4}x_4, x_2 = \frac{5}{4}x_4, x_3 = \frac{3}{4}x_4, \text{ 여기서 } x_4 \text{는 임의의 실수}$$

실제로 모든 화학방정식의 계수들은 정수이므로,  $x_4 = 4$ 로 잡으면  $x_1 = 1, x_2 = 5, x_3 = 3$ 이 된다. 따라서 균형 잡힌 화학방정식은

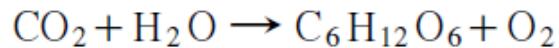


가 된다. 물론 각 계수들이 2배나 3배가 되더라도 상관이 없지만 가장 작은 정수를 선택한다. ■



## 예제 4-17

다음의 광합성 작용을 하는 화학방정식의 균형을 맞추어 보자.



**풀이**  $(x_1)\text{CO}_2 + (x_2)\text{H}_2\text{O} \rightarrow (x_3)\text{C}_6\text{H}_{12}\text{O}_6 + (x_4)\text{O}_2$ 가 균형 잡힌 화학방정식이 되도록 양의 정수  $x_1, x_2, x_3, x_4$ 를 구해야 한다. 먼저 각 원소의 수는 방정식의 양변에서 같아야 하므로

$$\text{탄소(C)} : x_1 = 6x_3$$

$$\text{수소(H)} : 2x_2 = 12x_3$$

$$\text{산소(O)} : 2x_1 + x_2 = 6x_3 + 2x_4$$

이것을 선형시스템으로 나타내면 다음과 같다.

$$x_1 - 6x_3 = 0$$

$$2x_2 - 12x_3 = 0$$

$$2x_1 + x_2 - 6x_3 - 2x_4 = 0$$

이것을 풀기 위해 가우스-조단 소거법을 적용하면 다음과 같다.

$$\left[ \begin{array}{cccc|c} 1 & 0 & -6 & 0 & 0 \\ 0 & 2 & -12 & 0 & 0 \\ 2 & 1 & -6 & -2 & 0 \end{array} \right]$$

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

$$\frac{1}{2} \times R_2 \rightarrow R_2$$

$$\left[ \begin{array}{cccc|c} 1 & 0 & -6 & 0 & 0 \\ 0 & 1 & -6 & 0 & 0 \\ 0 & 1 & 6 & -2 & 0 \end{array} \right]$$

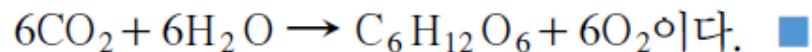
$$(-1) \times R_2 + R_3 \rightarrow R_3$$

$$\left[ \begin{array}{cccc|c} 1 & 0 & -6 & 0 & 0 \\ 0 & 1 & -6 & 0 & 0 \\ 0 & 0 & 12 & -2 & 0 \end{array} \right]$$

$$\frac{1}{12} \times R_3 \rightarrow R_3$$

$$\left[ \begin{array}{cccc|c} 1 & 0 & -6 & 0 & 0 \\ 0 & 1 & -6 & 0 & 0 \\ 0 & 0 & 6 & -1 & 0 \end{array} \right]$$

그 결과  $x_3 = \frac{1}{6}x_4$ ,  $x_2 = x_4$ ,  $x_1 = x_4$ 의 값들을 얻을 수 있다. 여기서는 모든 변수들이 정수가 되어야 하므로  $x_4 = 6$ 이라면,  $x_1 = 6$ ,  $x_2 = 6$ ,  $x_3 = 1$ ,  $x_4 = 6$ 이 된다. 따라서 구하고자 하는 최종 화학방정식은



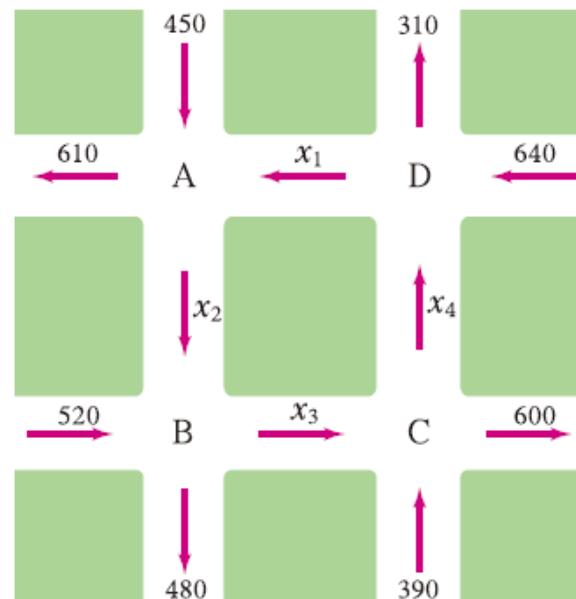
## 4.2.3 교통 흐름에의 응용





## 예제 4-18

어떤 도시의 중심가에 <그림 4.5>와 같은 4개의 일방통행 길이 있다고 한다. 각 교차점에 한 시간당 유입되는 교통량과 빠져나가는 교통량이 그림과 같이 주어졌을 경우 각 네거리에서의 교통량을 결정해 보자.



<그림 4.5> 4개의 일방통행 길

**풀이** 각 교차점에 유입되는 차량의 숫자와 빠져나가는 차량의 숫자가 같으므로, 교차점 A에 유입되는 차량의 수는  $x_1 + 450$ 이고 빠져나가는 차량의 수는  $x_2 + 610$ 이다.

따라서

$$x_1 + 450 = x_2 + 610 \quad (\text{교차점 A})$$

이와 같은 방법으로,

$$x_2 + 520 = x_3 + 480 \quad (\text{교차점 B})$$

$$x_3 + 390 = x_4 + 600 \quad (\text{교차점 C})$$

$$x_4 + 640 = x_1 + 310 \quad (\text{교차점 D})$$

과 같은 4개의 선형방정식을 만들 수 있다. 이것을 첨가행렬로 만들면 다음과 같다.

$$\left[ \begin{array}{cccc|c} 1 & -1 & 0 & 0 & 160 \\ 0 & 1 & -1 & 0 & -40 \\ 0 & 0 & 1 & -1 & 210 \\ -1 & 0 & 0 & 1 & -330 \end{array} \right]$$

이 행렬을 기약 행 사다리꼴로 변환시키면 다음과 같이 된다.

$$\left[ \begin{array}{cccc|c} 1 & 0 & 0 & -1 & 330 \\ 0 & 1 & 0 & -1 & 170 \\ 0 & 0 & 1 & -1 & 210 \\ 0 & 0 & 0 & 0 & 0 \end{array} \right]$$

이 시스템은 하나의 자유변수를 가지므로 여러 가지 해를 가질 수 있다. 만약 교차로 C와 D 사이의 평균 교통량이 한 시간당 200대라고 가정하면,  $x_4 = 200$ 일 것이고  $x_1, x_2, x_3$ 을  $x_4$ 에 대해 풀면 다음과 같다.

$$x_1 = x_4 + 330 = 530$$

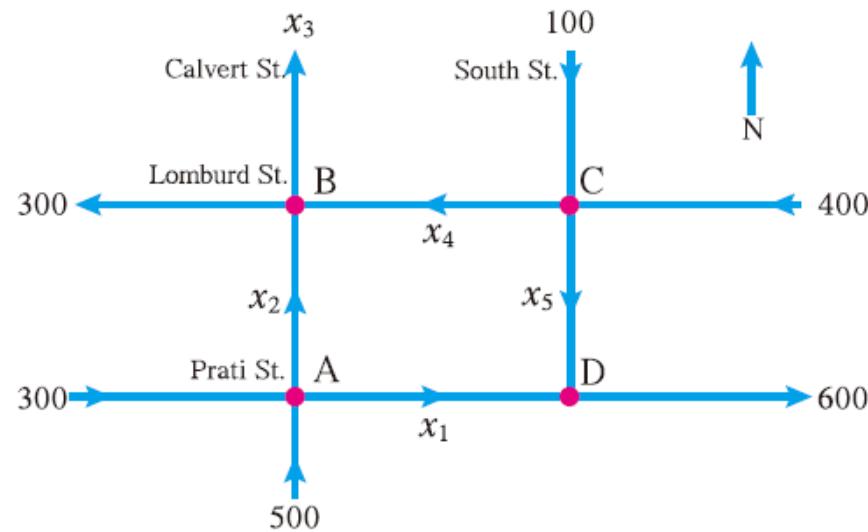
$$x_2 = x_4 + 170 = 370$$

$$x_3 = x_4 + 210 = 410 \quad \blacksquare$$



## 예제 4-19

〈그림 4.6〉의 네트워크는 미국의 어느 도시 일부의 오후 3시경 한 시간당 통과하는 자동차의 교통량을 나타낸 것이다. 이 네트워크의 일반적인 흐름 패턴을 결정해 보자.



〈그림 4.6〉 미국의 어느 도시의 도로 교통량

**풀이** 자동차의 흐름을 나타내는 선형방정식을 만들고 이 시스템의 일반 해를 구한다. 만약 각 교차로에서의 자동차의 유입량과 유출량이 같다고 가정하면 다음과 같은 식들을 얻는다.

교차로	유입량	=	유출량
A	$300 + 500$	=	$x_1 + x_2$
B	$x_2 + x_4$	=	$300 + x_3$
C	$100 + 400$	=	$x_4 + x_5$
D	$x_1 + x_5$	=	600

또한 네트워크상의 총 유입량( $500+300+100+400$ )은 총 유출량 ( $300+x_3+600$ )과 같으므로 이것을 간단히 하면  $x_3=400$ 이 나온다. 이 값과 처음 4개의 방정식을 결합하여 재정리하면 다음과 같은 선형시스템을 얻을 수 있다.

$$\begin{array}{rl}
 x_1 + x_2 & = 800 \\
 x_2 - x_3 + x_4 & = 300 \\
 & x_4 + x_5 = 500 \\
 x_1 & + x_5 = 600 \\
 x_3 & = 400
 \end{array}$$

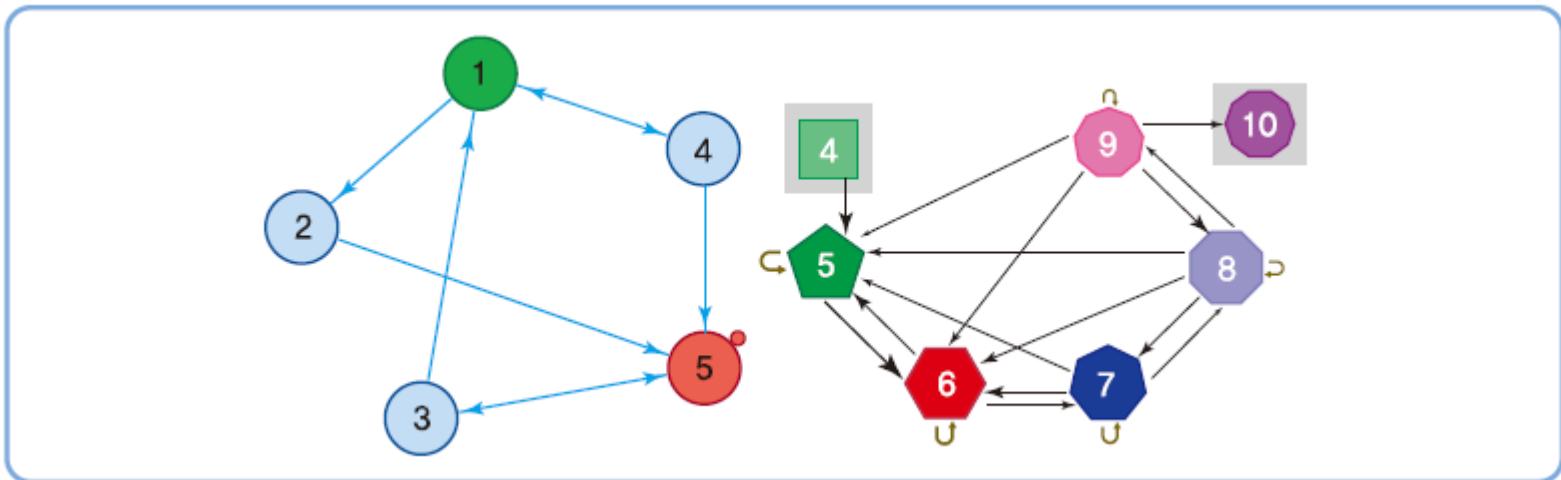
이것을 가우스 소거법으로 풀면 다음과 같다.

$$\begin{array}{rl}
 x_1 & + x_5 = 600 \\
 x_2 & - x_5 = 200 \\
 x_3 & = 400 \\
 x_4 + x_5 & = 500
 \end{array}$$

따라서 이 네트워크의 일반적인 흐름 패턴은 다음과 같이 구해진다.

$$\begin{cases} x_1 = 600 - x_5 \\ x_2 = 200 + x_5 \\ x_3 = 400 \\ x_4 = 500 - x_5 \\ x_5 \text{ 임의의 정수} \end{cases}$$

교차로에서의 음수 값은 사실상 그림에서의 반대 방향의 흐름으로 보면 된다. 이 문제에서 교통량은 음수가 있을 수 없으므로 위의 식에서  $x_4$ 가 음수가 될 수 없고  $x_5 \leq 500$ 이라는 제한을 가지게 되는 일반 해를 구할 수 있다. ■



마르코프 체인(Markov chain)은 연속적인 시간 간격에서의 상태벡터를 확률로 나타낸 것이다. 이때 행렬  $P$ 를 추이행렬(transition matrix)이라고 하며, 상태 벡터는 다음과 같은 방정식을 가진다.

$$\mathbf{x}(k+1) = P\mathbf{x}(k)$$



## 예제 4-20

어느 도시의 2010년도의 인구가 600,000명이었고, 그 인근 교외에 사는 인구가 400,000명이었다고 한다. 일 년 동안 도시에 사는 인구의 95%는 그 도시에 그대로 거주하고, 5% 정도의 인구는 인근 교외로 이사를 간다고 한다. 또한 그 도시의 인근 교외에 사는 인구의 3%는 도시로 이사를 가고 97%의 인구는 교외에 그대로 거주한다고 한다. 이 경우에 2011년과 2012년의 그 지역의 인구를 추정하여 계산해 보자.

**풀이** 주어진 조건에 따라 추이행렬  $P = \begin{bmatrix} .95 & .03 \\ .05 & .97 \end{bmatrix}$ 이며 2010년도의 처음 인구는  $x_0 = \begin{bmatrix} 600,000 \\ 400,000 \end{bmatrix}$ 으로 나타낼 수 있다. 2011년도에는 추이행렬에다 인구벡터를 곱하면  $x_1$ 의 결과와 같이 도시에 거주하는 인구는 582,000명이고, 인근 교외에 거

주하는 인구는 418,000명으로 추정할 수 있다.

$$\mathbf{x}_1 = \begin{bmatrix} .95 & .03 \\ .05 & .97 \end{bmatrix} \begin{bmatrix} 600,000 \\ 400,000 \end{bmatrix} = \begin{bmatrix} 582,000 \\ 418,000 \end{bmatrix}$$

그와 같은 방법을 적용하면 2012년도에는 다음과 같은 추정 인구를 구할 수 있다.

$$\mathbf{x}_2 = P\mathbf{x}_1 = \begin{bmatrix} .95 & .03 \\ .05 & .97 \end{bmatrix} \begin{bmatrix} 582,000 \\ 418,000 \end{bmatrix} = \begin{bmatrix} 565,440 \\ 434,560 \end{bmatrix} \blacksquare$$



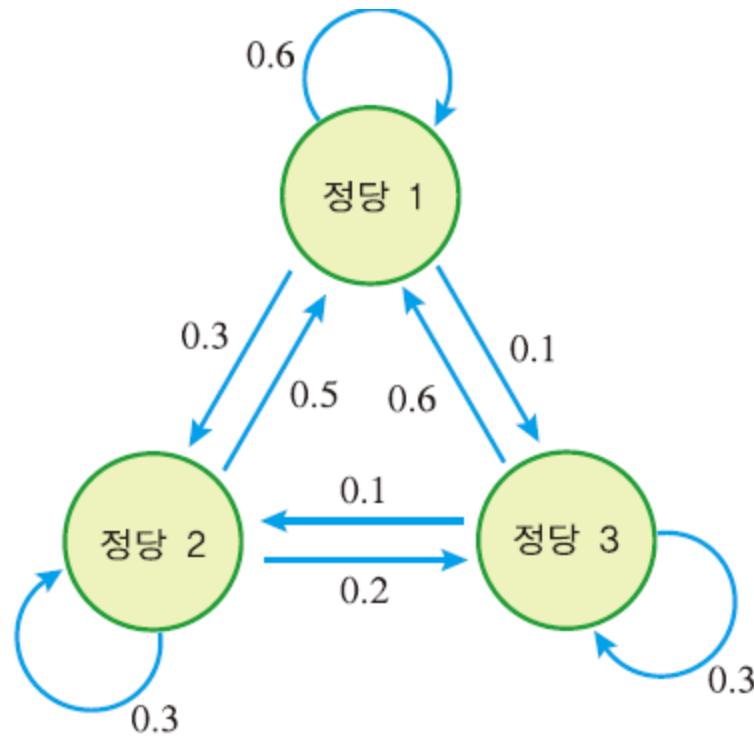
## 예제 4-21

마르코프 체인은 투표 성향에도 응용될 수 있는데, 어느 여론조사 기관에서 지지정당에 대한 조사를 주기적으로 실시할 때 유권자는 정당 1, 정당 2, 정당 3(무소속 포함)에 대해 지지 여부를 선택할 수 있다고 가정한다. 이러한 유권자의 지지 패턴은 <그림 4.7>과 같은 추이행렬  $P$ 를 가지는 마르코프 체인에 의해 모델링이 가능하다.

시각  $t = k$ 일 때의 지지 정당

$$P = \begin{bmatrix} 1 & 2 & 3 \\ 0.6 & 0.5 & 0.6 \\ 0.3 & 0.3 & 0.1 \\ 0.1 & 0.2 & 0.3 \end{bmatrix} \begin{array}{l} 1 \\ 2 \\ 3 \end{array}$$

시각  $t = k + 1$ 일 때의 지지 정당



〈그림 4.7〉 투표 성향을 나타내는 마르코프 모델

이것을 세부적으로 나타내면 다음과 같다.

$P_{11} = 0.6$  = 유권자가 정당 1을 계속하여 지지할 확률

$P_{12} = 0.3$  = 유권자가 정당 2에서 정당 1로 지지를 이동할 확률

$P_{13} = 0.1$  = 유권자가 정당 3에서 정당 1로 지지를 이동할 확률

$P_{21} = 0.5$  = 유권자가 정당 1에서 정당 2로 지지를 이동할 확률

$P_{22} = 0.3$  = 유권자가 정당 2를 계속 지지할 확률

$P_{23} = 0.2$  = 유권자가 정당 3에서 정당 2로 지지를 이동할 확률

$P_{31} = 0.6$  = 유권자가 정당 1에서 정당 3으로 지지를 이동할 확률

$P_{32} = 0.1$  = 유권자가 정당 2에서 정당 3으로 지지를 이동할 확률

$P_{33} = 0.3$  = 유권자가 정당 3을 계속 지지할 확률

$t$ 가 개월 수를 나타낸다고 할 때  $t = 0$ 일 때 유권자가 정당 1을 지지한다고 가정하고, 3개월 동안 유권자의 예상 지지 정당을 추정해 보자.

**풀이**  $x_1(k), x_2(k), x_3(k)$ 를 시각  $t = k$ 일 때 유권자가 각각 정당 1, 2, 3을 지지할 확률이라고 하고, 이때의 상태벡터를

$$\mathbf{x}(k) = \begin{bmatrix} x_1(k) \\ x_2(k) \\ x_3(k) \end{bmatrix}$$

라고 하자.  $t = 0$ 의 시각에 유권자가 정당 1을 지지하므로, 초기의 상태벡터는 다음과 같다.

$$\mathbf{x}(0) = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$$

$$\mathbf{x}(1) = P\mathbf{x}(0) = \begin{bmatrix} 0.6 & 0.5 & 0.6 \\ 0.3 & 0.3 & 0.1 \\ 0.1 & 0.2 & 0.3 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 0.6 \\ 0.3 \\ 0.1 \end{bmatrix}$$

3개월 동안의 상태벡터를 계산하면 다음과 같다.

$$\mathbf{x}(2) = P\mathbf{x}(1) = \begin{bmatrix} 0.6 & 0.5 & 0.6 \\ 0.3 & 0.3 & 0.1 \\ 0.1 & 0.2 & 0.3 \end{bmatrix} \begin{bmatrix} 0.6 \\ 0.3 \\ 0.1 \end{bmatrix} = \begin{bmatrix} 0.57 \\ 0.28 \\ 0.15 \end{bmatrix}$$

$$\mathbf{x}(3) = P\mathbf{x}(2) = \begin{bmatrix} 0.6 & 0.5 & 0.6 \\ 0.3 & 0.3 & 0.1 \\ 0.1 & 0.2 & 0.3 \end{bmatrix} \begin{bmatrix} 0.57 \\ 0.28 \\ 0.15 \end{bmatrix} = \begin{bmatrix} 0.572 \\ 0.270 \\ 0.158 \end{bmatrix}$$

$$\mathbf{x}(4) = P\mathbf{x}(3) = \begin{bmatrix} 0.6 & 0.5 & 0.6 \\ 0.3 & 0.3 & 0.1 \\ 0.1 & 0.2 & 0.3 \end{bmatrix} \begin{bmatrix} 0.572 \\ 0.270 \\ 0.158 \end{bmatrix} = \begin{bmatrix} 0.5730 \\ 0.2684 \\ 0.1586 \end{bmatrix}$$

그 결과 유권자가 정당 1을 지지할 확률은 57.30%, 유권자가 정당 2를 지지할 확률은 26.84%, 유권자가 정당 3을 지지할 확률은 15.86%로 추정할 수 있다. ■



## 예제 4-22

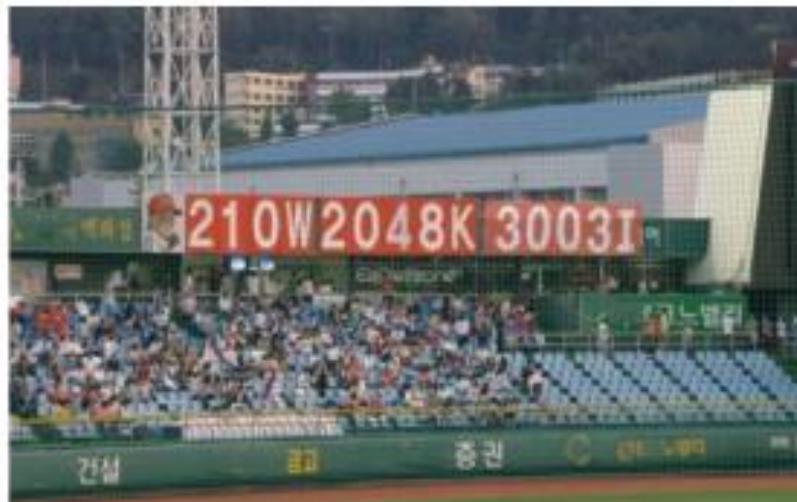
암호 해독에 있어서 코드화된 메시지를 주고받는 가장 일반적인 방법은 각 알파벳 문자마다 정수 값을 부여하고 그 메시지를 정수의 열(string)로 보내는 것이다 예를 들어, **SEND MONEY**라는 메시지는 5, 8, 10, 21, 7, 2, 10, 8, 3과 같이 코드화될 수 있다.

여기서 S는 5에 해당하고 Y는 3에 해당한다. 나머지도 순서에 따라 그 값을 가진다. 일반적으로 이런 형식의 문장은 해독하기가 비교적 쉬우므로 행렬의 곱을 이용하여 암호화하는 것이 좋다. 따라서 앞의 메시지를 다음 행렬  $A$ 와의 곱으로 변환시키면 해독하기가 매우 어렵게 될 것이다.

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

주어진 원래의 메시지(SEND MONEY)를 차례로 적으면 행렬  $B$ 와 같다.

## 4.2.5 암호 해독에의 응용



$$B = \begin{bmatrix} 5 & 21 & 10 \\ 8 & 7 & 8 \\ 10 & 2 & 3 \end{bmatrix}$$

암호화를 위해 행렬  $A$ 에다 원래의 메시지 행렬인  $B$ 를 곱하면

$$AB = \begin{bmatrix} 1 & 2 & 1 \\ 2 & 5 & 3 \\ 2 & 3 & 2 \end{bmatrix} \begin{bmatrix} 5 & 21 & 10 \\ 8 & 7 & 8 \\ 10 & 2 & 3 \end{bmatrix} = \begin{bmatrix} 31 & 37 & 29 \\ 80 & 83 & 69 \\ 54 & 67 & 50 \end{bmatrix}$$

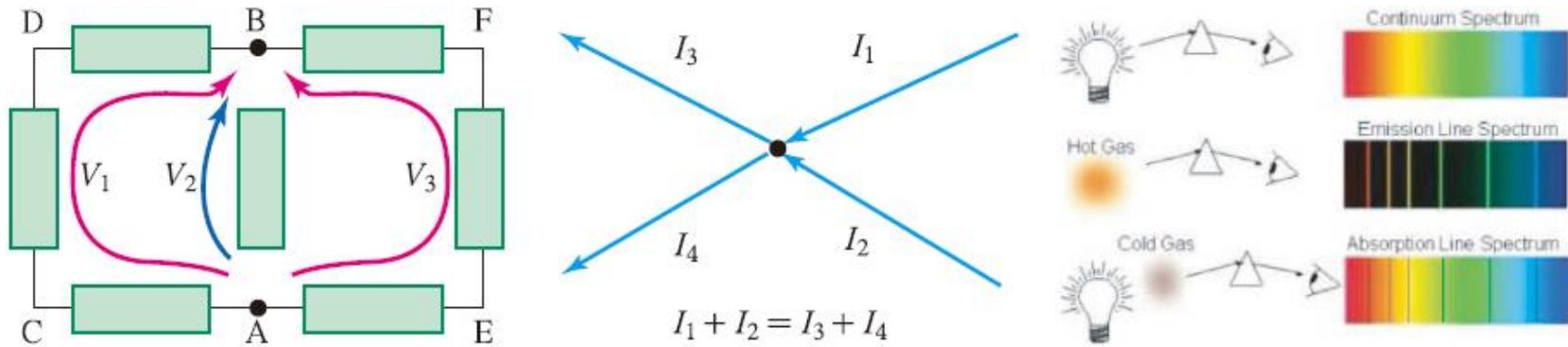
그러면  $A$ 와  $B$ 의 곱은 다음과 같이 코드화된 메시지로 나올 것이다.

31, 80, 54, 37, 83, 67, 29, 69, 50

그 메시지를 받는 사람은 받은 값의 행렬에다 미리 약속된 원래의 행렬  $A$ 의 역행렬인  $A^{-1}$ 를 곱함으로써 그 문자를 해독할 수 있다.

$$\begin{bmatrix} 1 & -1 & 1 \\ 2 & 0 & -1 \\ -4 & 1 & 1 \end{bmatrix} \begin{bmatrix} 31 & 37 & 29 \\ 80 & 83 & 69 \\ 54 & 67 & 50 \end{bmatrix} = \begin{bmatrix} 5 & 21 & 10 \\ 8 & 7 & 8 \\ 10 & 2 & 3 \end{bmatrix}$$

이 결과는 원래 보낸 행렬  $B$ 의 값과 같으므로 동일한 메시지로 해독하는 셈이다. ■

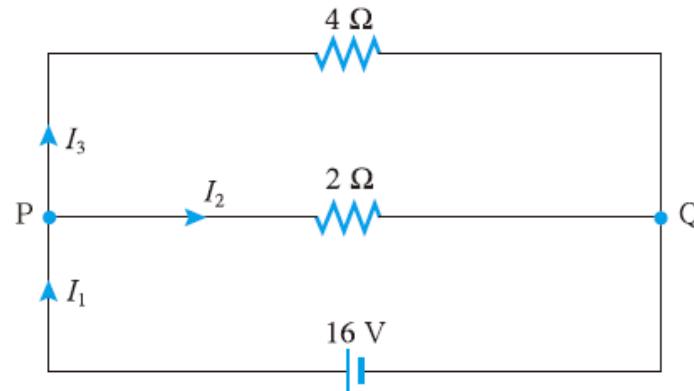


전류의 흐름이나 전압을 구하기 위한 방법으로 키르히호프(Kirchhoff)의 두 가지 법칙이 널리 쓰이고 있다. 회로상의 임의의 한 점으로 들어오는 전류의 합은 흘러나가는 전류의 합과 같다는 [키르히호프의 전류법칙](#)과 임의의 닫힌 회로에서 모든 전압강하의 합은 가해진 기전력과 같다는 [키르히호프의 전압법칙](#)을 이용한다.



## 예제 4-23

키르히호프의 법칙을 이용하여 <그림 4.8>과 같이 전기회로에 흐르는 전류의 값을 구해 보자.



<그림 4.8> 전기회로의 흐름

**풀이** 각 회로의 전류를  $I_1, I_2, I_3$ 이라고 하고 키르히호프의 법칙으로부터 전류에 관한 식을 구한다. P와 Q의 두 교차점을 중심으로 아래쪽의 닫힌 회로와 바깥의 닫힌 회로에서 다음과 같은 식을 얻을 수 있다.

$$I_1 - I_2 - I_3 = 0$$

$$2I_2 = 16$$

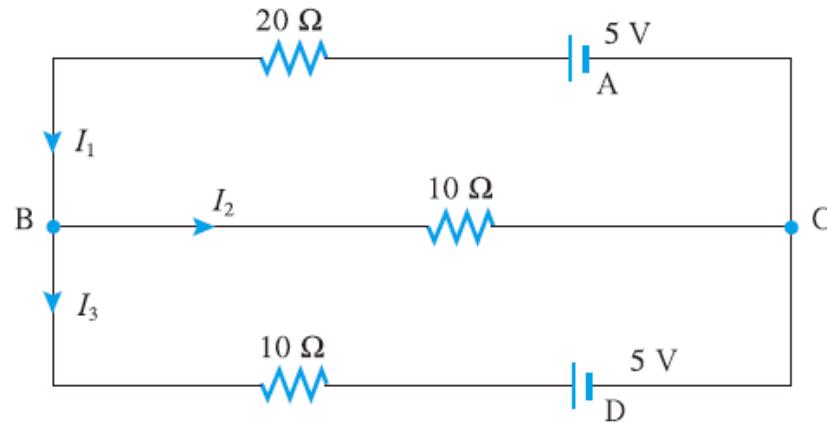
$$4I_3 = 16$$

두 번째 식으로부터  $I_2 = 8$ 을 얻고, 세 번째 식으로부터  $I_3 = 4$ 를 얻는다. 이 값을  
을 첫 번째 식에 대입하면  $I_1 = 12$ 를 얻는다.  
따라서 최종 해는  $I_1 = 12, I_2 = 8, I_3 = 4$ 이다. ■



## 예제 4-24

다음의 <그림 4.9>와 같은 전기회로에서 전류  $I_1$ ,  $I_2$ ,  $I_3$ 의 값을 결정해 보자.



<그림 4.9> 전기회로

**풀이** 키르히호프의 전류법칙을 교차점 B에 적용하면  $I_1 = I_2 + I_3$ 을 얻는다.  
따라서

$$I_1 - I_2 - I_3 = 0$$

## 선형방정식의 다양한 응용들

BCDB와 BCAB의 회로에서 키르히호프의 전압법칙을 사용하여 계산하면 다음과 같은 방정식을 얻는다.

$$10I_2 - 10I_3 = 5 \text{ (BCDB)}$$

$$20I_1 + 10I_2 = 5 \text{ (BCAB)}$$

이 세 방정식의 첨가행렬을 구하면 다음과 같다.

$$\left[ \begin{array}{ccc|c} 1 & -1 & 1 & 0 \\ 0 & 10 & -10 & 5 \\ 20 & 10 & 0 & 5 \end{array} \right]$$

이 행렬은 다음과 같은 기약 행 사다리꼴로 변형될 수 있다.

$$\left[ \begin{array}{ccc|c} 1 & -1 & -1 & 0 \\ 0 & 1 & -1 & 0.5 \\ 0 & 0 & 1 & -0.2 \end{array} \right]$$

따라서 변수의 값은 다음과 같다.

$$I_3 = -0.2$$

$$I_2 = I_3 + 0.5 = 0.3$$

$$I_1 = I_2 + I_3 = 0.1$$

여기서  $I_2$ 의 값이 음수이므로 전류의 흐름은 주어진 그림과는 달리 B로부터 D가 아니라 D로부터 B로 흐른다는 것도 알 수 있다. ■

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

#define MAX 5 // 행과 열의 최대값

// 선형방정식의 행렬 배열구조체 선언
typedef struct _matrix
{
    double a[MAX][MAX];
}matrix;

// 선형방정식 해의 행렬 배열구조체 선언
typedef struct _column
{
    double c[MAX];
}column;
```

```
// True, False 구조체 선언
typedef enum
{
    E_FALSE, E_TRUE
} E_BOOL;

void printout(matrix a, column c, int n);           // 연산 결과 출력함수
void gauss(matrix a, column c, int n);              // 가우스 소거법 계산함수
void backsub(matrix a, column c, int n);            // 역대입법 계산함수

// 연산결과를 출력한다.
void printout(matrix a, column c, int n)
{
```

```

int i = 0, j = 0;                                // 루프를 수행하기 위한 변수 선언

for(i = 0; i < n; i++)
{
    for(j = 0; j < n; j++)
    {
        printf(" %.1f\t", a.a[i][j]);
    }
    printf(" %d ", i+1);
    printf(" %.1f \n ", c.c[i]);
}
printf("\n");

// 입력받은 선형방정식을 가우스 소거법으로 계산한다.
void gauss(matrix a, column c, int n)
{
    E_BOOL error;

```

4.3

# C Program에 의한 선형방정식의 해법(가우스-조단 소거법)

```
int i = 0, j = 0, k = 0, l = 0;           // 루프를 수행하기 위한 변수 선언
double multi = 0, temp = 0;

printf("\n 가우스-조단소거법 풀이 과정: \n");
printf(" *****\n");

error = E_FALSE;
k = 0;

while(k < n && error != E_TRUE)
{
    l = k;
    for(j = k+1; j < n; j++)
    {
        if(fabs(a.a[j][k]) > fabs(a.a[l][k]))
        {
            l = j;
        }
    }
}
```

```
for(j = k; j < n; j++)
{
    temp = a.a[k][j];
    a.a[k][j] = a.a[1][j];
    a.a[1][j] = temp;
}
temp = c.c[k];
c.c[k] = c.c[1];
c.c[1] = temp;
if(a.a[k][k] != 0)
{
    for(j = k+1; j < n; j++)
    {
        multi = -1 * (a.a[j][k]) / a.a[k][k];
        for(i = k; i < n; i++)
        {
            if(multi != 0)
```

```
{  
    a.a[j][i] = a.a[j][i] + multi * a.a[k][i];  
}  
}  
c.c[j] = c.c[j] + multi * c.c[k];  
  
printout(a, c, n);  
}  
}  
else  
{  
    error = E_TRUE;  
}  
}  
k = k + 1;  
}  
  
if (error == E_TRUE)  
{
```

```

        printf("Trap condition.....");
    }

printout(a, c, n);
printf(" ****\n\n");
backsub(a, c, n);
}

// 역대입법 계산함수
void backsub(matrix a, column c, int n)
{
    int i = 0, j = 0, k = 0;           // 루프를 수행하기 위한 변수 선언
    float sum;
    column mat = {0.};

    printf(" a[%d, %d] = %.f\n\n", a.a[n-1][n-1]);
}

```

```
if(fabs(a.a[n-1][n-1]) == 0 || fabs(a.a[n-1][n-1]) < 1/1000000)
{
    printf(" This matrix is singular, does not have unique solution \n\n");
}
else
{
    mat.c[n-1] = c.c[n-1] / (a.a[n-1][n-1]);

    for(i = n-2; i > -1; i--)
    {
        sum = 0.0f;

        for(j = n-1; j > i; j--)
        {
            sum = sum + a.a[i][j] * mat.c[j];
        }

        if(fabs(a.a[n-1][n-1]) == 0 || fabs(a.a[n-1][n-1]) < 1/1000000)
```

```
{  
    printf(" This matrix is singular, does not have unique solution \n\n");  
}  
else  
{  
    mat.c[i] = (c.c[i] - sum) / a.a[i][i];  
}  
}  
  
printf(" 입력한 선형방정식의 해답\n");  
for(k = 0; k < n; k++)  
{  
    printf(" x%d = ", k+1);  
    printf("%.1f \t", mat.c[k]);  
}  
printf("\n\n");  
}
```

```
void main()
{
    int i = 0, j = 0; // 루프를 수행하기 위한 변수 선언
    int n; // 입력받을 행렬 값의 변수

    matrix a;
    column c;

    printf(" ****\n");
    printf(" **\n");
    printf(" **  가우스-조단 소거법을 이용한 선형방정식 계산 프로그램\n");
    printf(" **\n");
    printf(" ****\n\n");

    // 행렬(선형방정식의 차수)의 크기 값 입력
    printf(" 선형방정식의 최대 차수를 입력하세요: ");
}
```

```
scanf("%d", &n);

printf("\n");

// 선형방정식의 수식을 입력(A 행렬 각각의 값 입력)
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.a[i][j]);
    }
}
```

```
// 선형방정식의 결과 값을 입력(c 행렬의 값 입력)
printf("\n 선형방정식의 결과 값을 입력하세요. \n");
for(i = 0; i < n; i++)
{
    printf(" %d 번째 선형방정식의 결과 값: ", i+1);
    scanf("%lf", &c.c[i]);
}

printf("\n");

gauss(a, c, n);
}
```



## 실습 ④-1

## 예제 ①-4

C program

다음과 같은 2개의 변수를 가진 간단한 선형시스템을 구해 보자.

$$x_1 - 3x_2 = -3$$

$$2x_1 + x_2 = 8$$

이것을 풀면  $x_1 = 3$ ,  $x_2 = 2$ 가 주어진 선형시스템의 유일한 해가 된다. ■

```

C:\WINDOWS\system32\cmd.exe
*****
** 가우스-조단 소거법을 이용한 선형방정식 계산 프로그램 **
*****
선형방정식의 최대 차수를 입력하세요 : 2

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

선형방정식의 결과값을 입력하세요.
1 번째 선형방정식의 결과 값 : -3
2 번째 선형방정식의 결과 값 : 8

가우스 - 조단 소거법 풀이 과정 :
*****
+2      +1      x1  +8
+0      -4      x2  -7

+2      +1      x1  +8
+0      -4      x2  -7

*****
a[n, n] = -4

입력한 선형방정식의 해답
x1 = 3      x2 = 2

계속하려면 아무 키나 누르십시오 . . .

```



## 실습 4-2

## 예제 4-3

C program

다음 선형시스템의 해를 가우스-조단 소거법을 이용하여 구해 보자.

$$-x_2 - x_3 + x_4 = 0$$

$$x_1 + x_2 + x_3 + x_4 = 6$$

$$2x_1 + 4x_2 + x_3 - 2x_4 = -1$$

$$3x_1 + x_2 - 2x_3 + 2x_4 = 3$$

## 풀이

$$\left[ \begin{array}{cccc|c} 1 & 1 & 1 & 1 & 6 \\ 0 & -1 & -1 & 1 & 0 \\ 0 & 0 & -3 & -2 & -13 \\ 0 & 0 & 0 & -1 & -2 \end{array} \right]$$

그 결과 이 첨가행렬은 행 사다리꼴이 되고 역대입법에 의해  
 $x_1 = 2, x_2 = -1, x_3 = 3, x_4 = 2$ 인 해를 구할 수 있다. ■

```

C:\WINDOWS\system32\cmd.exe
*****
** 가우스-조단 소거법을 이용한 선형방정식 계산 프로그램 **
*****
선형방정식의 최대 차수를 입력하세요 : 4

선형방정식의 수식을 입력하세요.
1 번째 선형방정식 x1 의 값 : 0
1 번째 선형방정식 x2 의 값 : -1
1 번째 선형방정식 x3 의 값 : -1
1 번째 선형방정식 x4 의 값 : 1
2 번째 선형방정식 x1 의 값 : 1
2 번째 선형방정식 x2 의 값 : 1
2 번째 선형방정식 x3 의 값 : 1
2 번째 선형방정식 x4 의 값 : 1
3 번째 선형방정식 x1 의 값 : 2
3 번째 선형방정식 x2 의 값 : 4
3 번째 선형방정식 x3 의 값 : 1
3 번째 선형방정식 x4 의 값 : -2
4 번째 선형방정식 x1 의 값 : 3
4 번째 선형방정식 x2 의 값 : 1
4 번째 선형방정식 x3 의 값 : -2
4 번째 선형방정식 x4 의 값 : 2

선형방정식의 결과값을 입력하세요.
1 번째 선형방정식의 결과값 : 0
2 번째 선형방정식의 결과값 : 6
3 번째 선형방정식의 결과값 : -1
4 번째 선형방정식의 결과값 : 3

```

가우스 - 조단 소거법 풀이 과정 :

```
+1   +1   +1   +1   x1  +6
+0  -1  -1  +1   x2  +0
+2  +4  +1  -2   x3  -1
+3  +1  -2  +2   x4  +3
```

```
+1   +1   +1   +1   x1  +6
+0  -1  -1  +1   x2  +0
+0  +2  -1  -4   x3  -13
+3  +1  -2  +2   x4  +3
```

```
+1   +1   +1   +1   x1  +6
+0  -1  -1  +1   x2  +0
+0  +2  -1  -4   x3  -13
+0  -2  -5  -1   x4  -15
```

```
+1   +1   +1   +1   x1  +6
+0  +2  -1  -4   x2  -13
+0  +0  -2  -1   x3  -7
+0  +0  -6  -5   x4  -28
```

```
+1   +1   +1   +1   x1  +6
+0  +2  -1  -4   x2  -13
+0  +0  -6  -5   x3  -28
+0  +0  +0  +0   x4  +1
```

```
+1   +1   +1   +1   x1  +6
+0  +2  -1  -4   x2  -13
+0  +0  -6  -5   x3  -28
+0  +0  +0  +0   x4  +1
```

a[n, n] = 0

입력한 선형방정식의 해답

x1 = 2      x2 = -1      x3 = 3      x4 = 2

계속하려면 아무 키나 누르십시오 . . .



## 실습 4-3

## 예제 4-4

C program

다음의 선형시스템이 해를 가지는지를 판별해 보자.

$$\begin{aligned}x_1 - 2x_2 + x_3 &= 0 \\2x_2 - 8x_3 &= 8 \\-4x_1 + 5x_2 + 9x_3 &= -9\end{aligned}$$

**풀이** 삼각형 형태(triangular form)를 얻기 위해 필요한 행 연산을 하면 다음과 같다.

$$\begin{array}{l}x_1 - 2x_2 + x_3 = 0 \\x_2 - 4x_3 = 4 \\x_3 = 3\end{array} \quad \left[ \begin{array}{ccc|c}1 & -2 & 1 & 0 \\0 & 1 & -4 & 4 \\0 & 0 & 1 & 3\end{array} \right]$$

따라서 이 시스템은  $x_1 = 29$ ,  $x_2 = 16$ ,  $x_3 = 3$ 인 유일한 해를 가진다. ■

```

C:\WINDOWS\system32\cmd.exe
*****
** 가우스-조단 소거법을 이용한 선형방정식 계산 프로그램 **
*****
*****
```

선형방정식의 최대 차수를 입력하세요 : 3

선형방정식의 수식을 입력하세요.

```

1 번째 선형방정식 x1 의 값 : 1
1 번째 선형방정식 x2 의 값 : -2
1 번째 선형방정식 x3 의 값 : 1
2 번째 선형방정식 x1 의 값 : 0
2 번째 선형방정식 x2 의 값 : 2
2 번째 선형방정식 x3 의 값 : -8
3 번째 선형방정식 x1 의 값 : -4
3 번째 선형방정식 x2 의 값 : 5
3 번째 선형방정식 x3 의 값 : 9
```

선형방정식의 결과값을 입력하세요.

```

1 번째 선형방정식의 결과 값 : 0
2 번째 선형방정식의 결과 값 : 8
3 번째 선형방정식의 결과 값 : -9
```

가우스 - 조단 소거법 풀이 과정 :

$$\begin{array}{ccccc} +1 & -2 & +1 & \times 1 & +0 \\ +0 & +2 & -8 & \times 2 & +8 \\ -4 & +5 & +9 & \times 3 & -9 \end{array}$$

$$\begin{array}{ccccc} +1 & -2 & +1 & \times 1 & +0 \\ +0 & +2 & -8 & \times 2 & +8 \\ +0 & -3 & +13 & \times 3 & -9 \end{array}$$

$$\begin{array}{ccccc} +1 & -2 & +1 & \times 1 & +0 \\ +0 & -3 & +13 & \times 2 & -9 \\ +0 & +0 & +1 & \times 3 & +2 \end{array}$$

$$\begin{array}{ccccc} +1 & -2 & +1 & \times 1 & +0 \\ +0 & -3 & +13 & \times 2 & -9 \\ +0 & +0 & +1 & \times 3 & +2 \end{array}$$

`a[n, n] = 1`

입력한 선형방정식의 해답

$$x_1 = 29 \qquad x_2 = 16 \qquad x_3 = 3$$

계속하려면 아무 키나 누르십시오 . . .

# 선형방정식의 생활속의 응용

- 전기회로에서 키르히호프의 법칙을 이용하여 전류와 전압의 관계를 행렬로 나타냄으로써 그들의 관계를 명확히 구할 수 있다.
- 복잡한 화학방정식도 선형방정식을 이용하면 비교적 쉽게 풀 수 있다.
- 레온티에프의 경제 모델과 같이 경제학에도 선형방정식이 많이 응용되고 있다.
- 어떤 도로망에서 시간당 차량이 통과하는 수를 선형방정식으로 모델링함으로써 도로의 확장 등에 중요한 자료로 쓸 수 있다.
- 주어진 벡터에다 적절한 행렬을 곱함으로써 확대, 축소, 회전 등의 연산이 매우 편리하게 이루어지며, 이것을 여러 가지 응용에 널리 활용할 수 있다.
- 섭취하는 음식의 칼로리와 운동을 통해 소모되는 칼로리들의 관계를 행렬과 선형방정식으로 나타내어 풀 수 있으므로 전문적인 다이어트 관리에 응용될 수 있다.