3장

배열의 수학연산

###


LABORATORY
GREAT IDEA
EXPERIMENTS
DATA CENTER
RESEARCH CENTER
FIRST RESULTS

1. 배열의 덧셈과 뺄셈
2. 배열의 곱셈
3. 배열의 나눗셈
4. 원소별 연산
5. 내장함수에서의 배열 사용
6. 난수 발생
7. 응용 예제

개요

MATLAB의 배열 연산

- 배열 또는 스칼라 변수들의 수학 연산
  - 과학과 공학 분야의 응용문제에 대한 고급 배열 연산 가능
  - 가장 기본적 이며 일반적 수학 연산 학습
- 수학 연산 학습 내용
  - 배열의 덧셈과 뺄셈
  - 선형대수법칙에 따른 배열 연산의 곱셈, 나눗셈, 거듭제곱
    - *, /, ^ 이용
  - 원소별 연산(element-by-element) 연산의 곱셈, 나눗셈, 거듭제곱
    - .*, ./, .^ 이용

2.1 덧셈과 뺄셈

배열들 간의 덧셈과 뺄셈에
대하여 알아보자.

행렬의 덧셈과 뺄셈

덧셈과 뺄셈

❖ 같은 크기의(같은 행과 열의 수를 가진) 배열들에 대한 덧셈과 곱셈은 같은 위치의 원소들을 더하거나 빼서 구함.

$$\boldsymbol{A} = \begin{bmatrix} \boldsymbol{A}_{11} & \boldsymbol{A}_{12} & \boldsymbol{A}_{13} \\ \boldsymbol{A}_{21} & \boldsymbol{A}_{22} & \boldsymbol{A}_{23} \end{bmatrix} \quad \boldsymbol{B} = \begin{bmatrix} \boldsymbol{B}_{11} & \boldsymbol{B}_{12} & \boldsymbol{B}_{13} \\ \boldsymbol{B}_{21} & \boldsymbol{B}_{22} & \boldsymbol{B}_{23} \end{bmatrix} \Rightarrow \mathbf{A} + \mathbf{B} = \begin{bmatrix} (\boldsymbol{A}_{11} + \boldsymbol{B}_{11}) & (\boldsymbol{A}_{12} + \boldsymbol{B}_{12}) & (\boldsymbol{A}_{13} + \boldsymbol{B}_{13}) \\ (\boldsymbol{A}_{21} + \boldsymbol{B}_{21}) & (\boldsymbol{A}_{22} + \boldsymbol{B}_{22}) & (\boldsymbol{A}_{23} + \boldsymbol{B}_{23}) \end{bmatrix}$$

```
>> v = [3 6 2]; w=[9 -1  5];
>> z = v + w
z =
    12     5     7
>> A=[2 -3 7; 8  4  5];
>> B=[10 7 4; -11 15 1];
>> C = A - B
C =
     -8    -10     3
    19    -11     4
```

```
>> D = A + B
D = 12      4     11
      -3     19      6
>> D - 2
ans = 10      2      9
         -5     17      4
>> C * 2
ans =
   -16   -20     6
    38   -22     8
```

행렬과 스칼라의 연산

행렬과 스칼라의 덧셈, 뺄셈 등의 연산은 행렬의 **모든 원소에 스칼라**를 더하거나 뺀다.

MATLAB

배열의 덧셈과 뺄셈 응용예

예제

❖ 세 학생의 과목별 중간고사 및 기말고사 성적은 다음 표와 같다. 세 학생의 각 과목별 평균점수와 중간고사에 대한 성적향상 점수를 구하라.

이 름	중간고사				기말고사			
	국어	영어	수학	물리	국어	영어	수학	물리
김정은	61	78	39	42	69	84	51	46
강후동	49	57	24	36	55	53	31	40
이숭기	92	97	89	84	96	97	92	90

```
>> M = [61 78 39 42; 49 57 24 36; 92 97 89 84];  %중간고사 성적
>> F = [69 84 51 46; 55 53 31 40; 96 97 92 90];   %기말고사 성적
>> T = 0.5*(M + F)       % 학생별, 과목별 총점 구하기
T =    65.0000   81.0000   45.0000   44.0000
       52.0000   55.0000   27.5000   38.0000
       94.0000   97.0000   90.5000   87.0000
>> E = F – M       % 학생별, 과목별 성적 향상
E =    8      6     12      4
       6     -4      7      4
       4      0      3      6
```

3.2 행렬의 곱셈

배열들 간의 곱셈에 대하여 알아보자.

배열의 곱셈

행렬 곱셈의 규칙

❖ 행렬의 곱은 선형대수 법칙에 따름.

(**4**x3)(3x**2**) → (**4**x**2**)

$$A = \begin{bmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} \\ A_{41} & A_{42} & A_{43} \end{bmatrix} \qquad \mathrm{B} = \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \\ B_{31} & B_{32} \end{bmatrix}$$

$$A * B = \begin{bmatrix} (A_{11}B_{11} + A_{12}B_{21} + A_{13}B_{31}) & (A_{11}B_{12} + A_{12}B_{22} + A_{13}B_{32}) \\ (A_{21}B_{11} + A_{22}B_{21} + A_{23}B_{31}) & (A_{21}B_{12} + A_{22}B_{22} + A_{23}B_{32}) \\ (A_{31}B_{11} + A_{32}B_{21} + A_{33}B_{31}) & (A_{31}B_{12} + A_{32}B_{22} + A_{33}B_{32}) \\ (A_{41}B_{11} + A_{42}B_{21} + A_{43}B_{31}) & (A_{41}B_{12} + A_{42}B_{22} + A_{43}B_{42}) \end{bmatrix}$$

```
>> A = [ 1  4 3; 2  6  1; 5  2  8];
>> B = [5  4; 1  3; 2  6];
>> C = A*B
T =    15    34
       18    32
       43    74
>> D = B*A
??? Error using ==> *
Inner matrix dimensions must agree.
```

$$\begin{bmatrix} 1 & 4 & 3 \\ 2 & 6 & 1 \\ 5 & 2 & 8 \end{bmatrix}\begin{bmatrix} 5 & 4 \\ 1 & 3 \\ 2 & 6 \end{bmatrix} = \begin{bmatrix} (1\cdot5+4\cdot1+3\cdot2) & (1\cdot4+4\cdot3+3\cdot6) \\ (2\cdot5+6\cdot1+1\cdot2) & (2\cdot4+6\cdot3+1\cdot6) \\ (5\cdot5+2\cdot1+8\cdot2) & (5\cdot4+2\cdot3+8\cdot6) \end{bmatrix} = \begin{bmatrix} 15 & 34 \\ 18 & 32 \\ 43 & 74 \end{bmatrix}$$

$$A * B \neq B * A$$

행렬 곱과 벡터 곱

- 두 벡터dml; 곱 a x b
  - 두 벡터의 원소 개수 n으로 같아야 함.
  - a가 행벡터(1xn)이면 b는 열벡터(nx1)이어 함 → 스칼라 (1x1)
  - a가 열벡터 (nx1) 이면 b는 행벡터 (1xn) 이어야 함. → 행렬 (nxn)
  - 내장함수 dot(v, w) 는 두 벡터의 행·열 구분 없이 계산 가능.

$$\begin{bmatrix} a_1 & a_2 & a_3 \end{bmatrix}\begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} = a_1b_1 + a_2b_2 + a_2b_{21}, \qquad \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix}\begin{bmatrix} b_1 & b_2 & b_3 \end{bmatrix} = \begin{bmatrix} a_1b_1 & a_1b_2 & a_1b_3 \\ a_2b_1 & a_2b_2 & a_2b_3 \\ a_3b_1 & a_3b_2 & a_3b_3 \end{bmatrix}$$

```
>> a = [ 6   2  4 ]; b = [3;  9;  5];
>> x = a*b
x =
    56
>> z = b*a
z = 18     6     12
    54     18    36
    30     10    20
```

```
>> dot(a, b)          % a*b
ans =
      56
```

행렬 곱과 벡터 곱

❖ 행렬의 곱은 교환법칙이 성립하지 않음. 즉, **AB≠BA.**

```
>> A = [ 6   2; 4  7]; B = [3  9; 5  1];
>> C = A*B,  D = B*A     % A*B≠B*A
C =
    28    56
    47    43
D =
    54    69
    34    17
>> F=[1   3; 5   7]; G=[4   2; 1   6];
>> F*G
ans =
          7       20
         27       52
>> G*F
ans =
          14       26
          31       45
```

```
>> A=[2  5  7; 10  1  3; 6  2  11]
A =
     2     5     7
    10     1     3
     6     2    11
>> C = 3*A
C =
    6     15     21
   30      3      9
   18      6     33
>> D=A*3
 D =
    6     15      21
   30      3       9
   18      6      33
```

행렬과 스칼라의 연산

행렬과 스칼라의 곱셈, 나눗셈 등의 연산은 행렬의 **모든 원소에 스칼라**를 곱하거나 나눔.

연립선형방정식의 행렬 표현

연립방정식

❖ 연립선형방정식: 행렬로 간단히 표현하고 해를 체계적으로 구할 수 있음.

$$\begin{aligned} A_{11}x_1 + A_{12}x_2 + A_{13}x_3 &= B_1 \\ A_{21}x_1 + A_{22}x_2 + A_{23}x_3 &= B_2 \\ A_{31}x_1 + A_{32}x_2 + A_{33}x_3 &= B_3 \end{aligned}$$

해: $X = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$

❖ 행렬 표현

$$\begin{bmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} B_1 \\ B_2 \\ B_3 \end{bmatrix}$$

$$AX = B$$

❖ 해

$$X = A^{-1}B$$

▶ 단, A^{-1}는 의 A 역행렬

3.3 행렬의 나눗셈

배열들 간의 덧셈과 뺄셈에 대하여 알아보자.

연립선형방정식의 행렬 표현

단위 행렬

- 단위행렬(identity matrix or unity matrix) **I**
  - 대각선 원소가 모두 1이고 그 외의 원소는 0인 정방행렬(square matrix)
  - MATLAB에서는 **eye**(n) 명령어 사용
  - 행렬(또는 벡터)에 단위행렬 I를 곱해도 불변. (스칼라에 1을 곱한 경우와 유사)

$$AI = A, \qquad Iv = v, \qquad BI = B$$

$$\begin{bmatrix} 7 & 3 & 8 \\ 4 & 11 & 5 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 7 & 3 & 8 \\ 4 & 11 & 5 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 8 \\ 2 \\ 15 \end{bmatrix} = \begin{bmatrix} 8 \\ 2 \\ 15 \end{bmatrix} \qquad \begin{bmatrix} 6 & 2 & 9 \\ 1 & 8 & 3 \\ 7 & 4 & 5 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 6 & 2 & 9 \\ 1 & 8 & 3 \\ 7 & 4 & 5 \end{bmatrix}$$

  - 만일 A가 정방행렬이면

$$AI = IA = A$$

행렬의 역행렬(Inverse Matrix)

역행렬

- 정방행렬인 A와 B 에 대하여
  - **AB = BA = I** 를 만족하면 **A**와 **B**는 서로 역행렬
  - $A = IB^{-1} = B^{-1}$ 또는 $B = IA^{-1} = A^{-1}$
- MATLAB에서 A의 역행렬은 **A^−1** 또는 **inv(A)** 명령 사용

```
>> A= [ 7 4  6; 3  1  8; 2  5  4];
>> B=inv(A)
B =
     0.2278    -0.0886    -0.1646
    -0.0253    -0.1013     0.2405
    -0.0823     0.1709     0.0316
>> A*B   % B*A와 동일
ans =
     1.0000     0.0000     0.0000
          0     1.0000          0
    -0.0000     0.0000     1.0000
```

```
>> A^-1          % A의 역행렬
ans =
     0.2278    -0.0886    -0.1646
    -0.0253    -0.1013     0.2405
    -0.0823     0.1709     0.0316

>> A*A^-1      % AA-1=I
ans =
     1.0000     0.0000     0.0000
          0     1.0000          0
    -0.0000     0.0000     1.0000
```

행렬식(Determinant)

❖ 행렬 A의 행렬식 (determinant)

- |A|로 표시
- MATLAB 명령 **det(A)** 사용.

$$|\boldsymbol{A}| = \begin{vmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{vmatrix} = a_{11}a_{22} - a_{12}a_{21}$$

$$|\boldsymbol{B}| = \begin{vmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{vmatrix}$$

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

```
>> A= [ 1  2  0; 0  1  1; 3  0  1];  det(A)
ans =
       7
>> B=[ 1  2  1; 0  1  1; 3  1  1];  det(B)
ans =
       3
```

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

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

행렬 나눗셈

왼쪽 나누기 ₩

- 행렬의 나눗셈
  - **₩ :** 왼쪽 나눗셈 (left division)
  - **/ :** 오른쪽 나눗셈 (right division)
- **₩** : 왼쪽 나눗셈 (Left division)
  - 연립방정식 $\boldsymbol{Ax} = \boldsymbol{b}$ 의 해를 구하는 두가지 방법
    ① 역행렬로 구하기:
      - 양변에 $\boldsymbol{A}$의 역행렬 $\boldsymbol{A^{-1}}$을 곱하면,

        $$\boldsymbol{A^{-1}Ax} = \boldsymbol{A^{-1}b} \rightarrow \boldsymbol{x} = \boldsymbol{A^{-1}b}$$

      - MATLAB 명령어: **x = inv(A)*b**

    ② 왼쪽 나눗셈 **₩** 이용하기:
      - 가우스 소거법 (Gauss elimination) 이용
      - MATLAB 명령: **x = A₩b**
  - 차이점
    - 결과는 같지만 계산하는 방법이 서로 상이
    - 왼쪽 나눗셈 **₩**는 가우스 소거법으로 수치적으로 구함
    - 행렬의 크기가 크면 정확도 측면에서 가우스 소거법을 이용하는 왼쪽 나눗셈 **₩**이 역행렬 계산 보다 유리.

행렬 나눗셈

(2/2)

왼쪽 나눗셈

❖ ₩ : 왼쪽 나눗셈 예

$$\boldsymbol{Ax} = \boldsymbol{b} \qquad \begin{bmatrix} 4 & -2 & 6 \\ 2 & 8 & 2 \\ 6 & 10 & 3 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 8 \\ 4 \\ 0 \end{bmatrix}$$

```
>>  4/8            % 스칼라 오른쪽
ans =   0.5000
>> 4₩8              % 스칼라 왼쪽
ans =   2
>> A=[ 4   -2   6; 2   8   2; 6   10   3];
>> b=[8; 4; 0];
>> x=A^-1 * b       % 역행렬
ans =
   -1.8049
     0.2927
     2.6341
```

```
>>  x2 = inv(A)*b   % 역행렬
x2 =
   -1.8049
     0.2927
     2.6341
>> x3 = A₩b      % 왼쪽 나눗셈
x3 =
   -1.8049
     0.2927
     2.63412
```

MATLAB

행렬 나눗셈 (2/2)

오른쪽 나눗셈

❖ **/** : 오른쪽 나눗셈 (Right division)

▶ 연립 방정식 $xC = d$ 해를 구하는 두 가지 방법 (x와 d는 행벡터)

① 역행렬로 구하기:

- 양변에 $\boldsymbol{C}$의 역행렬 $\boldsymbol{C^{-1}}$을 곱하면,

$$\boldsymbol{xCC^{-1} = dC^{-1}} \rightarrow \boldsymbol{x = dC^{-1}}$$

- MATLAB 명령어: **x = b*inv(C)**

② 오른쪽 나눗셈 **/** 이용하기:

- 가우스 소거법 (Gauss elimination) 이용
- MATLAB 명령: **x = d/C**

```
>> C=[ 4  -2  6; 2  8  2; 6  10  3];
>> d=[8 4 0];
>> x1=d/C
ans =
   -1.8049    0.2927    2.6341
```

```
>>  x2 = d*inv(A)
x2 =
   -1.8049    0.2927    2.6341
```

과소 결정 연립방정식

underdetermined system

- 미지수의 수가 방정식의 수보다 많은 연립방정식
  - 이 경우 rank[A,b]가 미지수의 수보다 작다.
- pinv(A): pseudo inverse method
  - $A^T(AA^T)^{-1}$ 를 구하는 방법이며 근사해가 된다. (cf. $(A^TA)^{-1}A^T$)
- A : mxn 행열, b : mx1행열, x : nx1행열
  - m<n 일때 $Ax=b$의 해를 유도하라.

$$\boldsymbol{Ax} = \boldsymbol{b}$$ (mxn)(nx1)=(mx1)

양변에 의사 역행렬 $A^+ = A^T(AA^T)^{-1}$ (nxm)을 곱한다.

$$A^+Ax = A^T(AA^T)^{-1}Ax = A^T(AA^T)^{-1}b$$

$A^+A = A^T(AA^T)^{-1}A = \mathrm{I}$ 이므로,

$$x = A^T(AA^T)^{-1}b$$

과소 결정 연립방정식

예제

❖ 예

- $x_1+x_2+x_3=400,\ 10x_1+5x_2=1600$ 의 해를 구하라.
  - 미지수 3개 > 식수 2, 이므로 → 과소 결정 연립방정식
  - pinv 함수 이용

```
>> A = [1,1,1;10,5,0];
>> b = [400;1600];
>> x = pinv(A)*b
x =
   93.3333
  133.3333
  173.3333
```

```
>> PA=A'*inv(A*A');  %검증
>> x=PA*b
x =
   93.3333
  133.3333
  173.3333
```

과소 결정 연립방정식

underdetermined system

- 감소행 사다리꼴형(reduced row echelon form)이용
  - **Q=rref([A b])** 를 사용하여 Ax=b를 Cx=d 로 변형(Q=[C d])
  - 과소 결정 연립방정식에서 나머지 미지수의 함수를 몇몇 미지수로 표시
- 예
  - $x_1 + x_2 + x_3 = 400,\ 10x_1 + 5x_2 = 1600$ 의 해를 구하라.

```
>> A = [1,1,1;10,5,0];
>> b = [400;1600];
>> Q=rref([A b])
Q =
     1     0    -1   -80
     0     1     2   480
```

  - $x_1 = x_2 - 80,\ x_2 = 2x_3 + 480$ x_3로 x_1, x_2를 표시

과잉 결정 연립방정식

Overdetermined system

- 방정식의 수가 미지수의 수보다 많은 연립방정식
  - 이 경우 rank[A,b]가 미지수의 수보다 크다.
- 왼쪽나눗셈(₩)이용
  - 해는 최소 자승법(least squre method)의 의미를 갖는다.
  - 이 경우에는 **m>n** 이므로 pseudo-inverse를 $(A^TA)^{-1}A^T$ 로 사용.

$\boldsymbol{Ax = b}$, (mxn)(nx1)=(mx1)

양변에 의사 역행렬 $A^+ = (A^TA)^{-1}A^T$ (nxm)을 곱한다.

$$A^+Ax = (A^TA)^{-1}A^TAx = (A^TA)^{-1}A^Tb$$

$A^+A = (A^TA)^{-1}A^TA = \mathrm{I}$ 이므로,

$$x = (A^TA)^{-1}A^Tb$$

과잉 결정 연립방정식

Overdetermined system

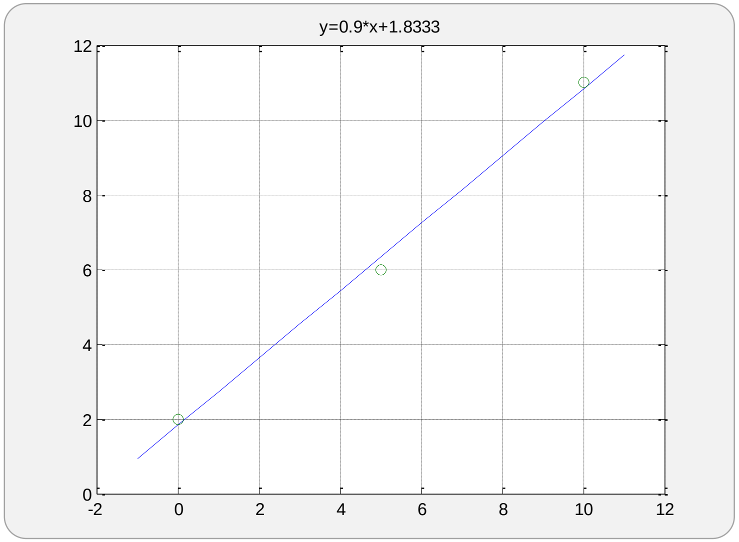
❖ 예(Curve fitting)

▶ 점 (0,2), (5,6), (10,11)을 가장 잘 표현하는 직선 $\mathbf{y=c_1x+c_2}$ 의 계수를 구하라.

$$\begin{bmatrix} 0 & 1 \\ 5 & 1 \\ 10 & 1 \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \end{bmatrix} = \begin{bmatrix} 2 \\ 6 \\ 11 \end{bmatrix}$$

```
>> A=[0 1; 5 1; 10 1]; b=[2; 6; 11];
>> x=A₩b
x =
    0.9000
    1.8333
>> pa=inv(A'*A)*A'        %검증
pa =
   -0.1000   -0.0000    0.1000
    0.8333    0.3333   -0.1667
>> x=pa*b
x =
    0.9000
    1.8333
```

MATLAB



3.4 원소별 연산

배열 원소 각각에 대해 실행되는 원소별 연산에 대하여 알아본다.

원소별(Element-by-Element) 연산

배열의 원소별 연산

- 원소별연산
  - 덧셈과 뺄셈 처럼 두 행렬의 원소와 원소끼리 곱셈 나누셈, 거듭제곱을 연산
  - 원소별 연산은 점 (.) 뒤에 연산기호를 붙이면 된다:

    .* (원소별 곱셈), .^ (원소별 거듭제곱), ./ (우측 나눗셈), .₩ (좌측 나눗셈)

기호	설명
.*	원소별 곱셈
.^	원소별 거듭제곱
./	원소별 오른쪽 나눗셈
. \	원소별 왼쪽 나눗셈

- 벡터의 원소별연산
  - **a**와 **b**는 각각 $\boldsymbol{a} = [\boldsymbol{a_1} \quad \boldsymbol{a_2} \quad \boldsymbol{a_3} \quad \boldsymbol{a_4}]$, $\boldsymbol{a} = [\boldsymbol{a_1} \quad \boldsymbol{a_2} \quad \boldsymbol{a_3} \quad \boldsymbol{a_4}]$일 때,

$$\boldsymbol{a}.*\ \boldsymbol{b} = [\boldsymbol{a_1 b_1} \quad \boldsymbol{a_2 b_2} \quad \boldsymbol{a_3 b_3} \quad \boldsymbol{a_4 b_4}]$$

원소별(Element-by-Element) 연산

행렬의 원소별 연산

❖ 행렬 **A** 와 **B** 의 원소별 연산

$$\mathbf{A} = \begin{bmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} B_{11} & B_{12} & B_{13} \\ B_{21} & B_{22} & B_{23} \\ B_{31} & B_{32} & B_{33} \end{bmatrix}$$

$$\mathbf{A.*\,B} = \begin{bmatrix} A_{11}B_{11} & A_{12}B_{12} & A_{13}B_{13} \\ A_{21}B_{21} & A_{22}B_{22} & A_{23}B_{23} \\ A_{31}B_{31} & A_{32}B_{32} & A_{33}B_{33} \end{bmatrix}$$

$$\mathbf{A./B} = \begin{bmatrix} A_{11}/B_{11} & A_{12}/B_{12} & A_{13}/B_{13} \\ A_{21}/B_{21} & A_{22}/B_{22} & A_{23}/B_{23} \\ A_{31}/B_{31} & A_{32}/B_{32} & A_{33}/B_{33} \end{bmatrix}$$

$$\mathbf{A.\,\hat{}\,n} = \begin{bmatrix} (A_{11})^n & (A_{12})^n & (A_{13})^n \\ (A_{21})^n & (A_{22})^n & (A_{23})^n \\ (A_{31})^n & (A_{32})^n & (A_{33})^n \end{bmatrix}$$

원소별(Element-by-Element) 연산

행렬의 원소별 연산

❖ 행렬 **A** 와 **B** 의 원소별 연산 예제

$$\mathbf{A} = \begin{bmatrix} 2 & 6 & 3 \\ 5 & 8 & 4 \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} 1 & 4 & 10 \\ 3 & 2 & 6 \end{bmatrix}$$

```
>> A=[2  6  3; 5  8  4];
>> B=[1 4 10; 3 2 7];
>> A.*B
ans = 2    24    30
      15    16    28
>> C=A./B
C =
    2.0000      1.5000      0.3000
    1.6667      4.0000      0.5714
```

```
>> 2.^B
ans =
    2        16        1024
    8         4         128
>> A*B
??? Error using==> mtimes
Inner  matrix  dimensions  must
   agree.
```

원소별(Element-by-Element) 연산

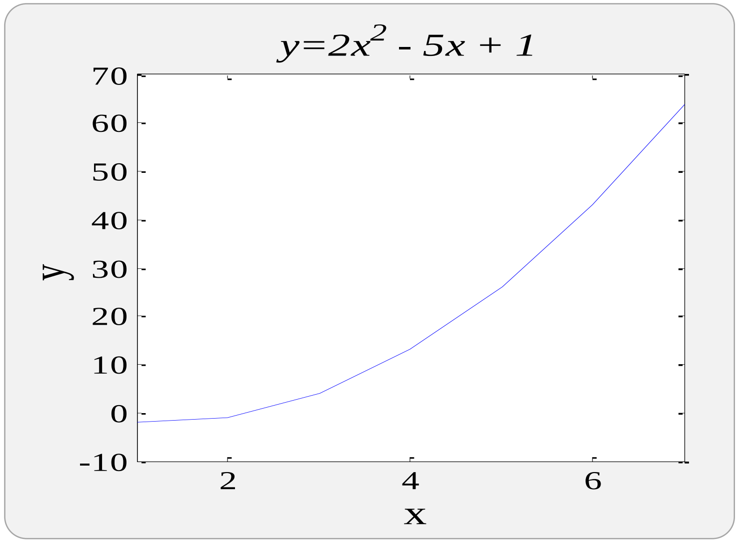
행렬의 원소별 연산의 예

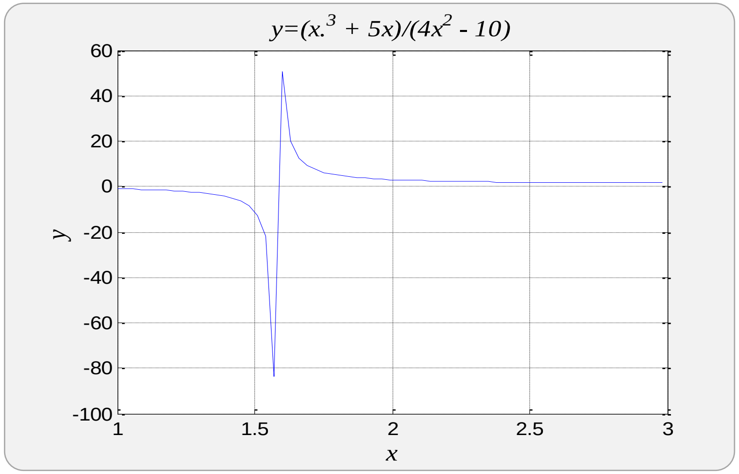
- 원소별 연산을 이용한 함수 그래프 그리기
  - 독립변수 x 를 정의
  - 함수값 구하기
- $y(x) = 2x^2 + 5x + 1$ 함수. $(1 \le x \le 7)$

```
>> x = 1:7;         % 독립변수 정의
>> y = (2*x.^2 – 5*x + 1); % 함수
>> plot(x, y)         % 그래프 그리기
```

- $y(x) = \frac{x^3+5x}{4x^2-10}$ 함수. $(1 \le x \le 3)$

```
>> x = [1:0.03:3]; % 독립변수 정의
>> y = (x.^3 + 5*x)./(4*x.^2 - 10);
>> plot(x, y)       % 그래프 그리기
>> grid
```



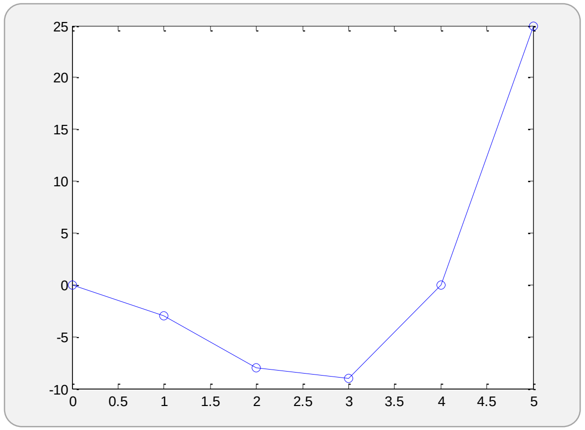


원소별(Element by Element) 연산

행렬의 원소별 연산의 예

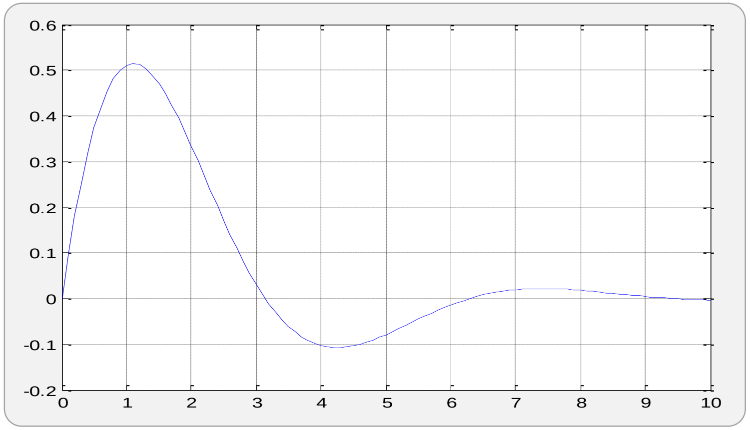
- 그래프를 그릴 때 독립변수와 종속변수를 모두 벡터로 간주.
- $\boldsymbol{y} = \boldsymbol{x}^3 - 4\boldsymbol{x}^2$의 그래프를 $\boldsymbol{x} = [\mathbf{0}, \mathbf{5}]$에서 그려라.

```
>> x=[0:5]; y=x.^3-4*x.^2
y =
     0     -3     -8     -9     0     25
>> plot(x,y,'-o')
```



- $\boldsymbol{y} = \boldsymbol{e}^{-0.5\boldsymbol{x}} \sin \boldsymbol{x}$, $\boldsymbol{x} = [\mathbf{0}, \mathbf{10}]$.

```
>> x=[0:0.1:10];y=exp(-0.5*x).*sin(x);
>> plot(x,y)
>> grid
```



MATLAB

3.5 내장함수에서 의 배열 사용

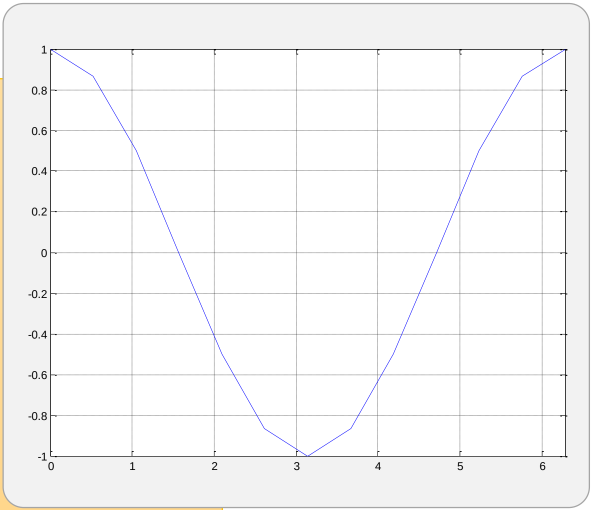
내장함수에 입력인자가 배열인 경우와 내장 배열해석 함수에 대하여 알아본다.

내장함수에서의 배열 사용

내장함수 인자가 배열인 경우

- 내장함수의 입력인자가 배열인 경우:
  - 함수에 의해 정의된 연산이 원소별 연산처럼 배열의 각 원소에 대해 수행됨.
  - 출력은 입력 배열과 같은 크기의 배열이 됨.
  - 예: x가 배열인 경우 **cos**(x) 함수의 계산

```
>> x=[0:pi/6:2*pi]
x =
   Columns 1 through 7
   0    0.5236    1.0472    1.5708    2.0944    2.6180    3.1416
   Columns 8 through 13
   3.6652    4.1888    4.7124    5.2360    5.7596    6.2832
>> y=cos(x)
y =
   Columns 1 through 7
   1.0000    0.8660    0.5000    0.0000    -0.5000    -0.8660    -1.0000
   Columns 8 through 13
   -0.8660    -0.5000    -0.0000    0.5000    0.8660    1.0000
>> plot(x, y), grid on
```



내장함수에서의 배열 사용

내장함수 인자가 배열인 경우

- 예: x가 배열인 경우 **sqrt** (x) 함수의 계산

```
>> d=[1  4  9; 16  25  36; 49  64  81]
d =
     1     4     9
    16    25    36
    49    64    81
>> h=sqrt(d)
h =
     1     2     3
     4     5     6
     7     8     9
```

$$\mathbf{A} = \begin{bmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} \end{bmatrix}$$

$$sqrt(\mathbf{A}) = \begin{bmatrix} \sqrt{A_{11}} & \sqrt{A_{11}} & \sqrt{A_{11}} \\ \sqrt{A_{11}} & \sqrt{A_{11}} & \sqrt{A_{11}} \\ \sqrt{A_{11}} & \sqrt{A_{11}} & \sqrt{A_{11}} \end{bmatrix}$$

배열 해석용 내장함수

내장 배열 해석함수

함수	설명	예
mean(A)	**A**가 벡터이면, 벡터 원소들의 평균값을 돌려준다.	>> A = [5 9 2 4]; >> mean(A) ans = 5
C = max(A) **[d,n]= max(A)**	**A**가 벡터이면, **C**는 **A**에서 가장 큰 원소이다. **A**가 행렬일 경우, **C**는 **A**의 각 열에서 가장 큰 원소들로 구성된 행벡터이다. **A**가 벡터이면, **d**는 **A**에서 가장 큰 원소이며 **n**은 원소의 위치이다(최대값이 여러 개인 경우에는 첫 번째 최대값의 위치).	>> A = [5 9 2 4 11 6 11 1]; >> C = max(A) C = 11 >>[d,n] = max(A) d = 11 n = 5
min(A) **[d,n] = min(A)**	최소값을 찾는다는 점을 제외하고는 max (A)와 동일하다. 최소값을 찾는다는 점을 제외하고는 [d,n] = max(A)와 동일하다.	>> A = [5 9 2 4]; >> min(A) ans = 2

배열 해석용 내장함수

내장 배열 해석함수

함수	설명	예
sum(A)	**A**가 벡터이면, 벡터 원소들의 합을 돌려준다.	>> A = [5 9 2 4]; >> sum(A) ans = 20
sort(A)	**A**가 벡터이면, 벡터의 원소들을 오름차순으로 정렬한다.	>> A = [5 9 2 4]; >> sort(A) ans = 2 4 5 9
median(A)	**A**가 벡터이면, 벡터 원소들의 중앙값(median value)을 돌려준다.	>> A = [5 9 2 4]; >> median(A) ans = 4.5000
std(A)	**A**가 벡터이면, 벡터 원소들의 표준편차를 돌려준다.	>> A = [5 9 2 4]; >> std(A) ans = 2.9439

배열 해석용 내장함수

내장 배열 해석함수

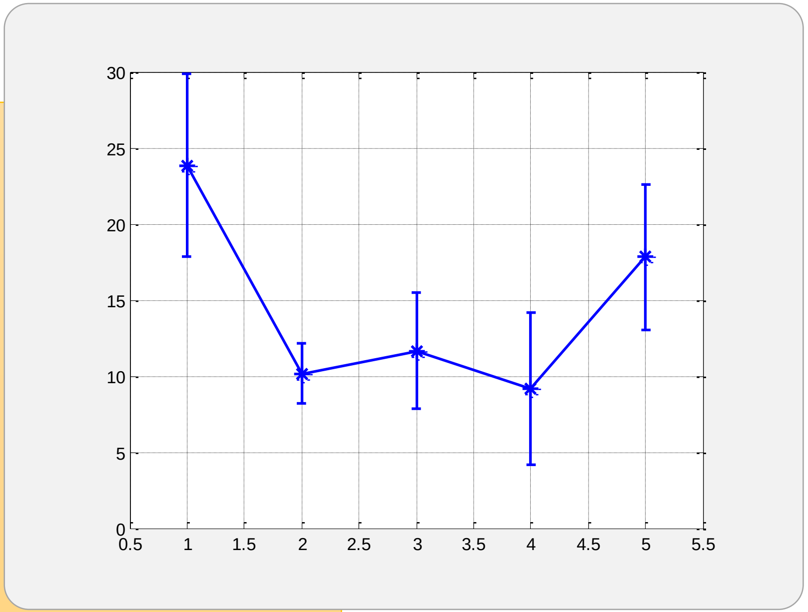
함수	설명	예
det(A)	정방행렬 **A**의 행렬식을 돌려준다.	>> A = [5 9 2 4]; >> det(A) ans = -2
dot(a,b)	두 벡터 **a**와 **b**의 스칼라 곱(내적)을 계산한다. 벡터는 각각 행벡터 또는 열벡터가 될 수 있다.	>> a = [1 2 3]; >> b = [2 4 1]; >>dot(a,b) ans = 26
cross(a,b)	두 벡터 **a**와 **b**의 외적(cross product)을 계산한다. 두 벡터는 **3**개의 원소를 가져야한다.	>> a = [1 2 3]; >> b = [2 4 1]; >>cross(a,b) ans = -5 3 -2
inv(A)	정방행렬 **A**의 역행렬을 돌려준다.	>> A = [2 -2 1; 3 2 -1; 2 -3 2]; >> inv(A) ans = 0.2000 0.2000 0 -1.6000 0.4000 1.0000 -2.6000 0.4000 2.0000

배열 해석용 내장함수

내장 배열 해석함수

❖ 주어진 배열에 대하여 mean과 std 를 계산하고 errorbar() 함수를 이용하여 그래프로 표시

```
figure
 h =[ 24    10    11     7    19
      15     7     9     7    24
      21    12     7     4    19
      27    11    13     7    15
      33    12    12    12    10
      23     9    18    18    20]
means = mean(h);     % 평균
stds = std(h);        % 표준편차
errorbar(means, stds, 'b*-', 'MarkerSize', 10, 'LineWidth', 2);
grid
```



3.7 난수의 발생

공학적 문제에 필요한 다양한 난수(random number) 발생 방법을 알아본다.

난수의 생성

- 공학 응용에서 난수들이 필요.
  - 내장함수로 rand(), randn() 지원.
  - **rand** 명령 :
    - 0과 1 사이에서 균일하게 분포된 난수 생성
    - rand : 한 개의 난수 생성, rand(1, n) : n개의 난수 행벡터 생성
    - rand(n) : nxn 크기의 난수 행렬 생성, rand(m, n) : mxn 크기의 난수행렬 생성
  - **randperm(n)** 명령 :
    - 1에서 n까지의 정수의 무작위 순열(permutation)으로 구성된 1x n의 행벡터 생성
    - >> x = randperm(8) ➔ x = 8 2 7 4 3 6 5 1
  - **randn** 명령:
    - 평균이 0 이고 표준편차가 1인 정규분포의 난수 생성
- a~b 사이의 난수 생성
  - (b-a)*rand+a
- 1~n 사이의 정수 난수 생성 :
  - randi(n) 명령
  - round((n-1)*rand+1)

임의 평균과 표준편차의 정규분포

$x = randn$: $(\mu_x, \sigma_x) = (0, 1)$

$y = ax + b$ 를 적용 →

$(\mu_y, \sigma_y) = (a\mu_x + b, a\sigma_x)$

난수의 발생

❖ randi 명령: 정수 난수 생성.

명령어	설명	예
randi(imax) **(imax 는 정수)**	**1**과 **imax** 사이의 난수 정수한 개를 생성한다.	>>a = randi(15) a = 9
randi(imax, n)	**1**과 **imax** 사이의 난수 정수들로 구성된 **n x n** 행렬을 생성한다.	>>b = randi(15,3) b = 4 8 11 14 3 8 1 15 8
randi(imax, m, n)	**1**과 **imax** 사이의 난수 정수들로 구성된 **m x n** 행렬을 생성한다.	>>c = randi(15,2,4) c = 1 1 8 13 11 2 2 13

```
d =randi([50 90], 3, 4)
A = 64    77    80    81
    59    65    73    52
    64    71    84    72
```

난수의 생성

❖예) 1에서 100 사이의 정수 난수를 갖는 2×15 행렬 생성

```
A=round(99*rand(2,10)+1)
A = 44    77    20    45    71    28    66    13    96    59
    39    80    49    65    76    68    17    50    35    23
```

❖ 예) randn 명령어 :

```
>> d=randn(3,4)
d =
    -0.5883   -0.1364    1.0668
     2.1832    0.1139    0.0593
>> x = round( 5*randn(1, 10) + 10)        % 평균 10 표준편차 5
x =  6    4    10    10    7    10    10    3    13    8
```

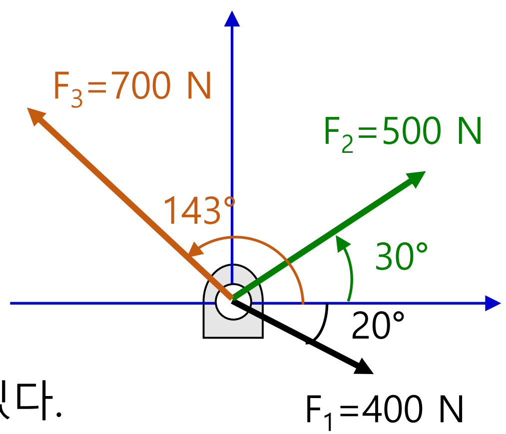
3.7 응용예제

다양한 공학 응용 예제를 통하여 이해의 깊이를 더한다.

예제 3.2 등가 힘의 계산

등가힘 계산

❖ 그림과 같이 고리에 가해지는 세 힘의 합력(등가 힘)을 구하라.



▶ 풀이

- 직교 좌표계에서 힘 벡터 F는 다음과 같이 쓸 수 있다.
- $\boldsymbol{F} = F_x\hat{\boldsymbol{\imath}} + F_y\hat{\boldsymbol{\jmath}} = F\cos\theta\,\hat{\boldsymbol{\imath}} + F\sin\theta\,\hat{\boldsymbol{\jmath}} = F(\cos\theta\,\hat{\boldsymbol{\imath}} + \sin\theta\,\hat{\boldsymbol{\jmath}})$
- 여기서 $F = \sqrt{F_x^2 + F_y^2}$ 이고 $\tan\theta = \frac{F_y}{F_x}$

❖ 합력(등가 힘)은 다음 세 단계에 따라 구함.

① 각 힘을 x 축 성분,y 축 성분으로 나타낸다.

② 세 벡터를 더하여 등가 힘을 벡터 형태로 구한다.

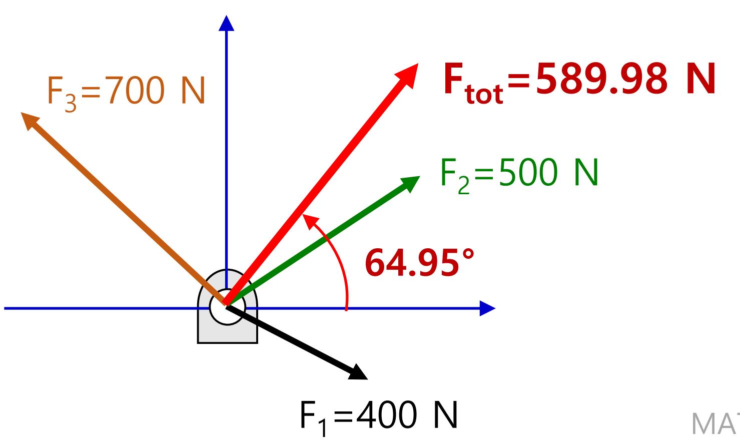
③ 등가 힘의 크기와 방향을 구한다.

예제 3.2 등가 힘의 계산 (2/2)

❖ 스크립트파일로 작성

```
clear
F1M=400;F2M=500; F3M=700;            % 벡터의 크기
th1=-20; th2=30; th3=143;            % 벡터의 각도
F1=F1M*[cosd(th1) sind(th1)]         % 벡터 F1
F2=F2M*[cosd(th2) sind(th2)]
F3=F3M*[cosd(th3) sind(th3)]
Ftot=F1+F2+F3                         % 합벡터
FtotM=sqrt(Ftot(1)^2+Ftot(2)^2)       % 합벡터 크기
Th=atand(Ftot(2)/Ftot(1))             % 합벡터 각도
```

```
>> format compact
>> Ch3_1
F1 =
  375.8770 -136.8081
F2 =
  433.0127  250.0000
F3 =
 -559.0449  421.2705
Ftot =
  249.8449  534.4625
FtotM =
  589.9768
Th =
   64.9453
```

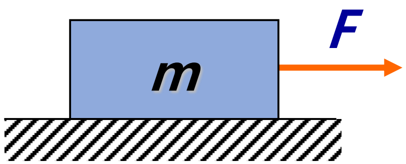


예제 3.3 마찰 실험

원소별 계산

- 실험에서 마찰 계수 μ는 질량 m을 이동시키는데 필요한 힘 F를 측정하면 다음 식에 의해 구할 수 있다.

$$\mu = \frac{F}{mg}, \quad \text{여기서, } g = 9.81\, m/s^2$$



- 여섯 번의 시험을 통한 F 에 대한 측정 결과가 다음 표로 주어졌다. 각 시험에 대한 마찰계수를 구하고, 모든 시험 결과에 대한 마찰계수의 평균을 구하라.

측정	1	2	3	4	5	6
m (kg)	2	4	5	10	20	50
F (N)	12.5	23.5	30	61	117	294

```
>>  m=[2  4  5   10   20   50];  F = [12.5,  23.5, 30, 61, 117, 294];
>>  mu=F./(m*9.81)
mu = 0.6371    0.5989    0.6116    0.6218    0.5963    0.5994
>> mu_ave = mean(mu)
mu_ave =  0.6109
```

예제 3.4 전기저항회로 분석 (1/3)

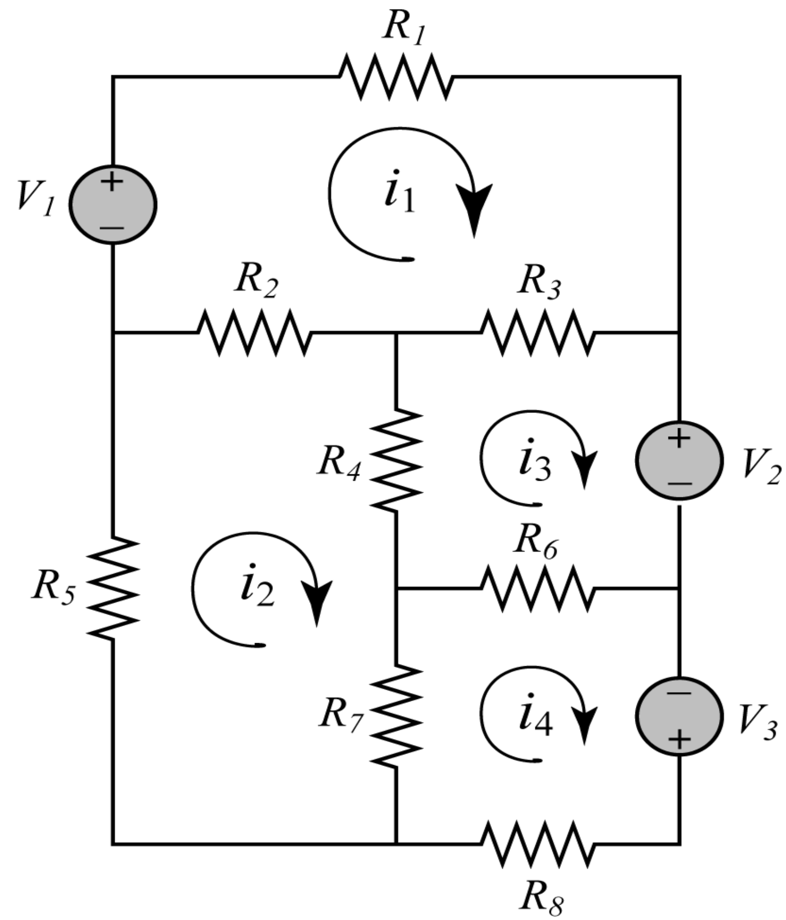
❖ 다음 전기회로에 대해 Kirchhoff의 전압법칙에 의한 망전류 방법 (mesh current method)을 이용하여 각 저항에서의 전류를 구하라.

V1 = 20 V, V2 = 12 V, V3 = 40 V

R1 = 18 Ω, R2 = 10 Ω, R3 = 16 Ω

R4 = 6 Ω, R5 = 15 Ω, R6 = 8 Ω

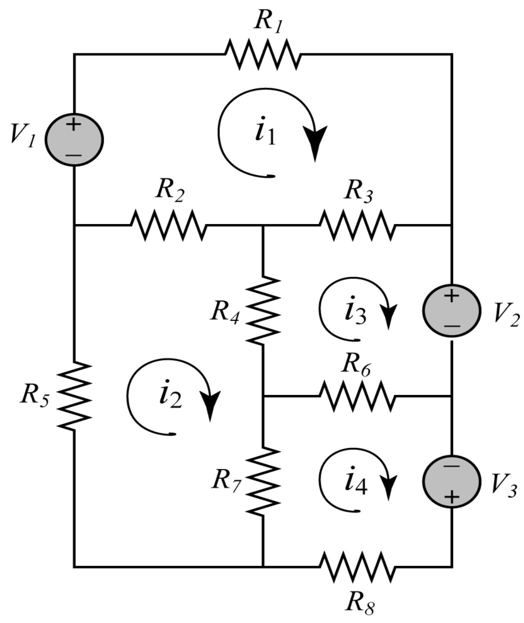
R7 = 12 Ω, R8 = 14 Ω



예제 3.4 전기저항회로 분석

❖ 풀이 :

- Kirchhoff 전압법칙(voltage law) : 폐회로상의 전압을 모두 더하면 합이 0
- 각 망에 전류 i_1, i_2, i_3, i_4 를 할당한 후, Kirchhoff 전압 제2법칙을 각 망에 적용하면, 전류에 대한 선형 연립방정식 4개를 얻을 수 있다.
- 이 연립방정식으로부터 망 전류의 i1, i2, i3, i4 값을 구할 수 있다.
- 두 망에 모두 속해 있는 저항을 흐르는 전류는 각 망에서의 전류를 더한 값과 같다.
- 모든 전류는 시계방향으로 흐른다고 가정한다.
- 각 망에 대한 식에서, 전압원의 부호는 전류가 -극 쪽으로 흐를 때 양(+)이며, 저항의 전압 부호는 망전류 방향의 전류에 대해 음(-)이다.



예제 3.4 전기저항회로 분석

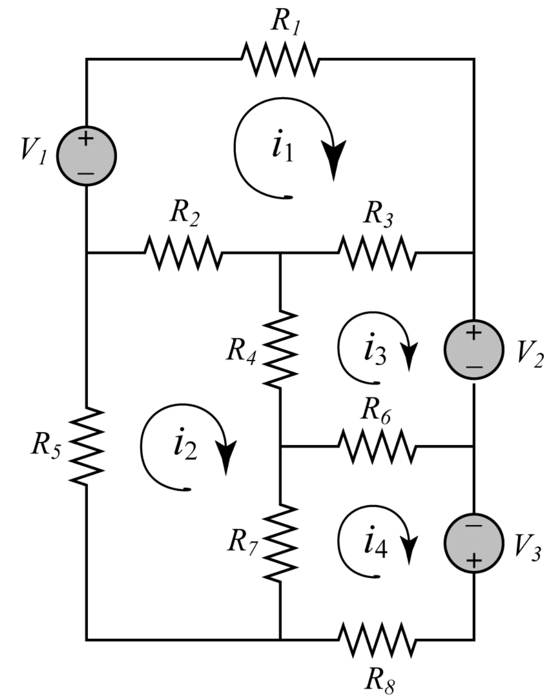
❖ 네 망에 대해 전압 제2법칙을 적용하여 얻은 연립방정식

$$\boldsymbol{V_1 - R_1 i_1 - R_3(i_1 - i_3) - R_2(i_1 - i_2) = 0}$$

$$\boldsymbol{-R_5 i_2 - R_2(i_2 - i_1) - R_4(i_2 - i_3) - R_7(i_2 - i_4) = 0}$$

$$\boldsymbol{-V_2 - R_6(i_3 - i_4) - R_4(i_3 - i_2) - R_3(i_3 - i_1) = 0}$$

$$\boldsymbol{V_3 - R_8 i_4 - R_7(i_4 - i_2) - R_6(i_4 - i_3) = 0}$$



❖ 행렬 형태로 나타낸 연립방정식

$$\begin{bmatrix} (\boldsymbol{R}_1 + \boldsymbol{R}_2 + \boldsymbol{R}_3) & -\boldsymbol{R}_2 & -\boldsymbol{R}_3 & 0 \\ \boldsymbol{R}_2 & -(\boldsymbol{R}_2 + \boldsymbol{R}_4 + \boldsymbol{R}_5 + \boldsymbol{R}_7) & \boldsymbol{R}_4 & \boldsymbol{R}_7 \\ \boldsymbol{R}_3 & \boldsymbol{R}_4 & -(\boldsymbol{R}_3 + \boldsymbol{R}_4 + \boldsymbol{R}_6) & \boldsymbol{R}_6 \\ 0 & \boldsymbol{R}_7 & \boldsymbol{R}_6 & -(\boldsymbol{R}_6 + \boldsymbol{R}_7 + \boldsymbol{R}_8) \end{bmatrix} \begin{bmatrix} \boldsymbol{i}_1 \\ \boldsymbol{i}_2 \\ \boldsymbol{i}_3 \\ \boldsymbol{i}_4 \end{bmatrix} = \begin{bmatrix} \boldsymbol{V}_1 \\ 0 \\ \boldsymbol{V}_2 \\ -\boldsymbol{V}_3 \end{bmatrix}$$

예제 3.4 전기저항회로 분석

스크립트 및 실행 결과

❖ 스크립트와 실행결과

```
V1=20;V2=12;V3=40;
R1=18; R2=10; R3=16; R4=6;
R5=15; R6=8;  R7=12; R8=14;
A=[(R1+R2+R3), -R2, -R3, 0;
    R2, -(R2+R4+R5+R7), R4, R7;
    R3, R4, -(R3+R4+R6), R6;
    0, R7, R6, -(R6+R7+R8)]
B=[V1;0;V2;-V3]
I=A₩B
```

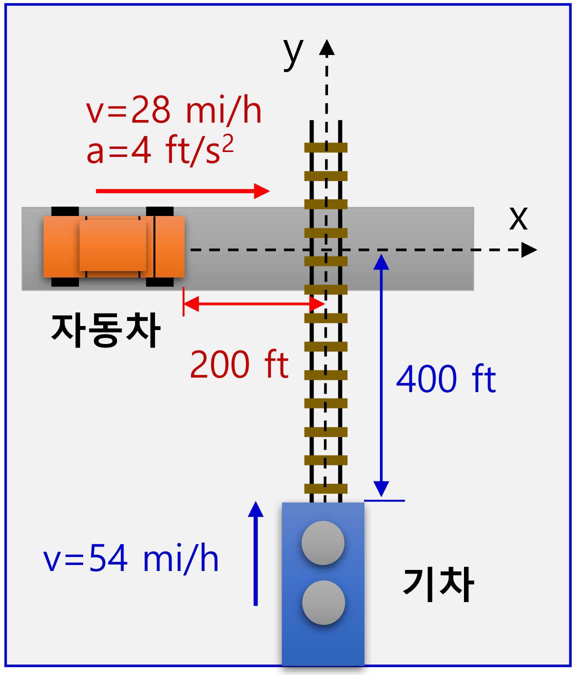
```
>> ch3_4
A =
   44   -10   -16     0
   10   -43     6    12
   16     6   -30     8
    0    12     8   -34
B =
   20
    0
   12
  -40
I =
    0.8411
    0.7206
    0.6127
    1.5750
```

$$\begin{bmatrix} i_1 \\ i_2 \\ i_3 \\ i_4 \end{bmatrix}$$

예제 3.5 두 질점의 운동

기차와 자동차

- 기차 (t=0)
  - 위치 : 교차로의 남쪽 400 ft 지점
  - 속도: 54 mi/h 북쪽으로 정속
- 자동차 (t=0)
  - 위치 : 교차로의 서쪽 200 ft 지점
  - 속도: 동쪽으로 28 mi/h
  - 가속도: 동쪽으로 4 ft/s2
- 이후 10초 동안에 대해 매초마다
  - (1) 시간, (2) 기차의 위치, (3) 자동차의 위치, (4)기차와 자동차 사이의 거리, (5) 자동차의 속력, (6) 자동차에 대한 기차의 상대 속력
  - 결과는 11×6의 행렬로 생성하여 출력한다.



1 mi/h = 5280/3600 ft/s

예제 3.5 두 질점의 운동

기차와 자동차

❖ 풀이 :

▶ 위치

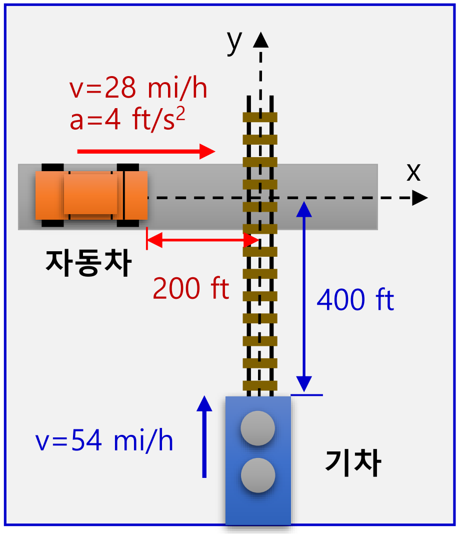
- 등가속운동의 일반공식: 직선을 따라 움직이는 물체의 위치

$$s = s_0 + v_o t + \frac{1}{2} a t^2$$

- 일정한 가속도로 직선을 따라 움직이는 기차와 자동차의 위치:

기차 : $y = -400 + v_{0trian} t$

자동차 : $x = -200 + v_{ocar} t + \frac{1}{2} a_{car} t^2$



- 기차와 자동차 사이의 거리: $d = \sqrt{x^2 + y^2}$

▶ 속도

- 기차의 속도 (등속 +y 방향)

$$v_{train} = v_{train} \hat{\jmath}$$

- 자동차 속도 (등가속 + x방향)

$$v_{car} = (v_{car0} + a_{car} t)\hat{\imath}$$

- 상대속도 (자동차에서본 기차의 속도)

$$v_{train/car} = v_{train} - v_{car} = -(v_{car0} + a_{car} t)\hat{\imath} + v_{train} \hat{\jmath}$$

예제 3.5 두 질점의 운동 (2/3)

기차와 자동차

❖ 스크립트 실행 결과

```
%
v0train=54*5280/3600; v0car=28*5280/3600; acar=4;  % 초기 속도
t=0:10;
y=-400+ v0train*t;
x=-200+ v0car*t +0.5*acar*t.^2;
d=sqrt(x.^2+y.^2);        % 상대 거리
vcar=v0car+acar*t;
speed_trainRcar=sqrt(vcar.^2+v0train.^2);
table=[t', y', x', vcar', speed_trainRcar']
```

```
>> ch3_5
table =
```

시간	기차위치	자동차 위치	자동차속력	상대속력
0	-400.0000	-200.0000	41.0667	89.2139
1.0000	-320.8000	-156.9333	45.0667	91.1243
2.0000	-241.6000	-109.8667	49.0667	93.1675
3.0000	-162.4000	-58.8000	53.0667	95.3347
4.0000	-83.2000	-3.7333	57.0667	97.6178
5.0000	-4.0000	55.3333	61.0667	100.0089
6.0000	75.2000	118.4000	65.0667	102.5003
7.0000	154.4000	185.4667	69.0667	105.0849
8.0000	233.6000	256.5333	73.0667	107.7561
9.0000	312.8000	331.6000	77.0667	110.5075
10.0000	392.0000	410.6667	81.0667	113.3333

Thank you!

3 장 끝