MATLAB으로 배우는 공학 수치해석(3판)

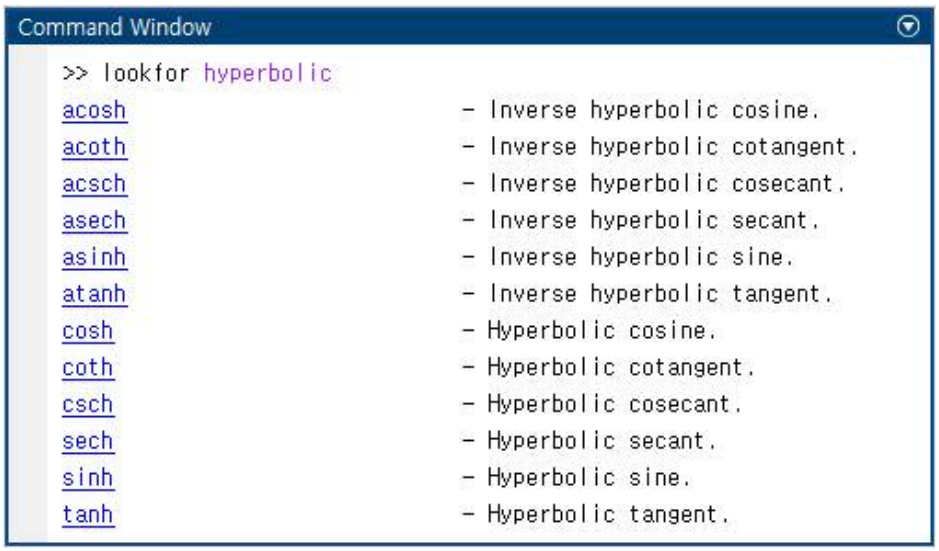
[연습문제 정답 이용 안내]

- 본 연습문제 정답의 저작권은 방성완과 한빛아카데미(주)에 있습니다.
- **이 자료를 무단으로 전제하거나 배포할 경우 저작권법 136조에 의거하여 최고 5년 이하의 징역 또는 5천만원 이하의 벌금에 처할 수 있고 이를 병과(倂科)할 수도 있습니다.**

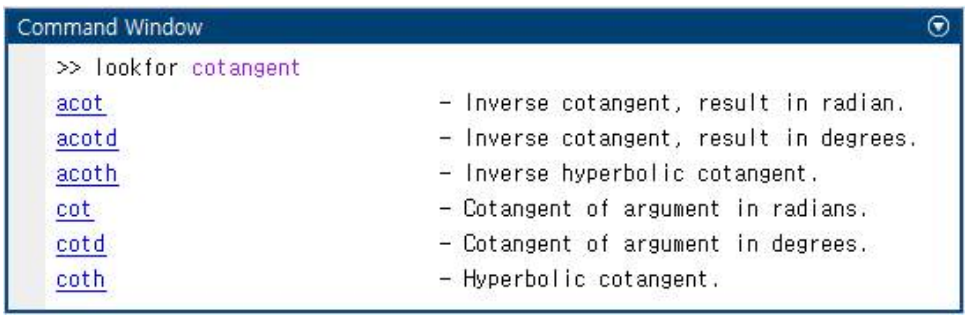
Chapter 01 연습문제

1.1

다음과 같이 입력하고 Enter 키를 누르면 'hyperbolic' 또는 'cotangent' 철자가 포함된 함수가 명령창에 나타난다.



또는 다음과 같이 입력하고 Enter 키를 눌러도 함수가 명령창에 나타난다.



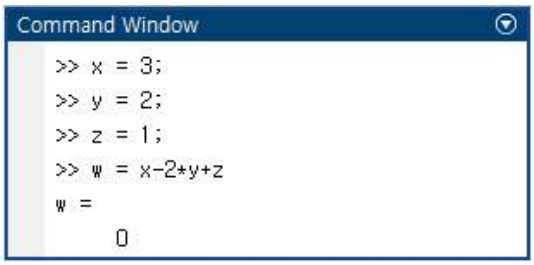
1.2

```
>> x = 0;
>> y = exp(-2*x) - 2*cos(x) + 10^-4;
```

또는

```
>> x = 0;
>> y = exp(-2*x) - 2*cos(x) + 1e^-4;
```

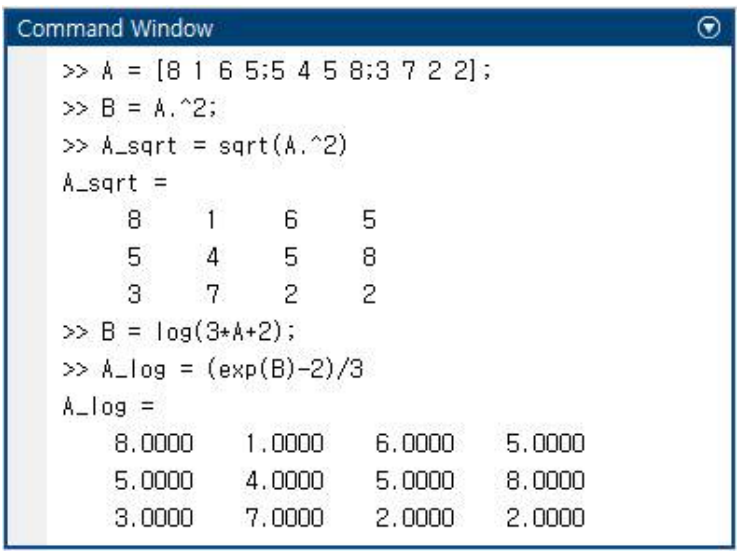
1.3



1.4

>>y = [sin(2*pi),sin(pi/6),sin(pi/2), 2]'

1.5



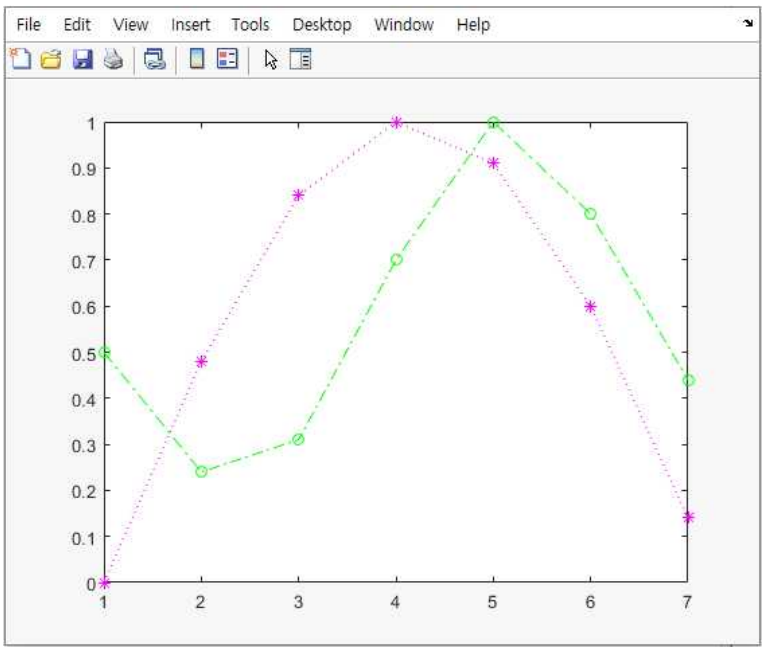
배열 A를 지정하고 배열 연산자를 이용하여 각 원소의 제곱을 계산한다. 로그와 지수의 역 연산을 이용하면 원래 배열 A를 복구할 수 있다.

1.6

```
Command Window
>> u = [5,6,3;2,7,7;1,3,9];
>> x = u(2:3,[1,3])
x =
     2     7
     1     9
>> y = u(:,2)
y =
     6
     7
     3
>> u(1,:) = [ ]
u =
     2     7     7
     1     3     9
```

1.7

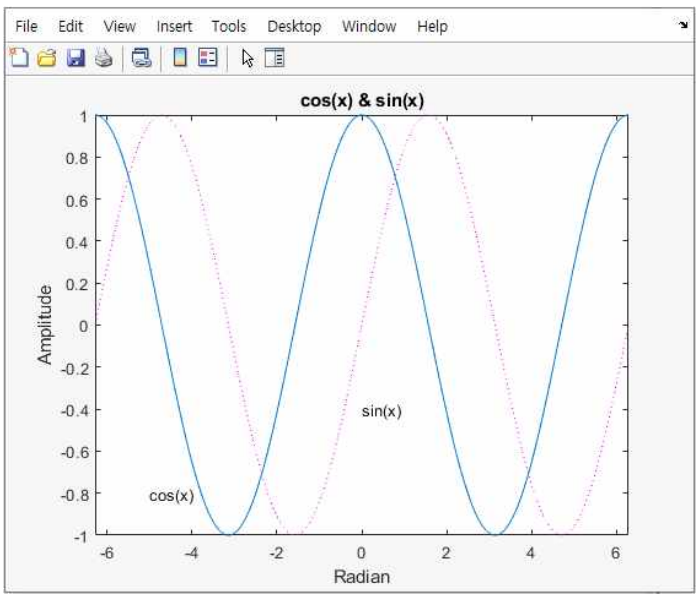
```
Command Window
>> x = [1 2 3 4 5 6 7];
>> y1 = [0 0.48 0.84 1 0.91 0.6 0.14];
>> y2 = [.5 .24 .31 .7 1 .8 .44];
>> plot(x,y1,':*m',x,y2,'-.og');
```



1.8

```
Command Window
>> x = -2*pi:pi/360:2*pi;
>> y1 = cos(x);
>> y2 = sin(x);
>> plot(x,y1,x,y2,':m');
>> axis([-2*pi 2*pi -1 1]);
>> title('cos(x) & sin(x)');
>> xlabel('Radian');
>> ylabel('Amplitude');
>> text(0.0,-0.4,'sin(x)');
>> text(-5.0,-0.8,'cos(x)');
```

xlabel, ylabel, title 명령어는 상호 순서를 고려할 필요가 없다.



text 명령어는 순서를 고려하지 않고 지정할 수 있다. 그러나 text 명령어를 사용하는 경우 그래프에 문자열이 삽입되는 최적화 위치를 예상해야 하는 번거로움이 있다.

1.9

관계 연산자의 오른쪽과 왼쪽의 변수 혹은 상수를 비교하여 조건을 만족하는 원소에는 참값을 표시하는 “1”을, 만족하지 않는 경우는 거짓값을 표시하는 “0”을 표시하게 된다.

```
Command Window
>> x = [9,5,-1,-2,4];
>> y = [8,1,-2,-1,4];
>> z1 = (x >= 1)
z1 =
  1×5 logical array
   1   1   0   0   1
>> z2 = (x <= y)
z2 =
  1×5 logical array
   0   0   0   1   1
>> z3 = (x == y)
z3 =
  1×5 logical array
   0   0   0   0   1
>> z4 = (x ~= y)
z4 =
  1×5 logical array
   1   1   1   1   0
```

1.10

```
Command Window
>> x = [-7,-3,12,3,8,0];
>> y = [-7,-2,10,3,6,-1];
>> z1 = ~(x&y)
z1 =
  1×6 logical array
   0   0   0   0   0   1
>> z2 = x|y
z2 =
  1×6 logical array
   1   1   1   1   1   1
>> z3 = xor(y,x)
z3 =
  1×6 logical array
   0   0   0   0   0   1
```

1.11

(a) x1과 x2를 비교하여 x1이 x2보다 큰 값의 원소 자리에 해당하는 x1의 실제 원소 값을 표시한다.
(b) (a)에서 x1의 원소값에 해당하는 원소의 자리 번호를 나타낸다.
(c) x1과 x2를 비교하여 x1이 x2보다 작거나 같은 값의 원소 자리에 해당하는 x2의 실제 원소값을 표시한다.

Command Window

```
>> x1 = [1,4,7,-2,0,0];
>> x2 = [9,2,3,0,-1,-2];
>> z1 = x1(x1 > x2)
z1 =
     4     7     0     0
>> z2 = find(x1 > x2)
z2 =
     2     3     5     6
>> z3 = x2(x1 <= x2)
z3 =
     9     0
```

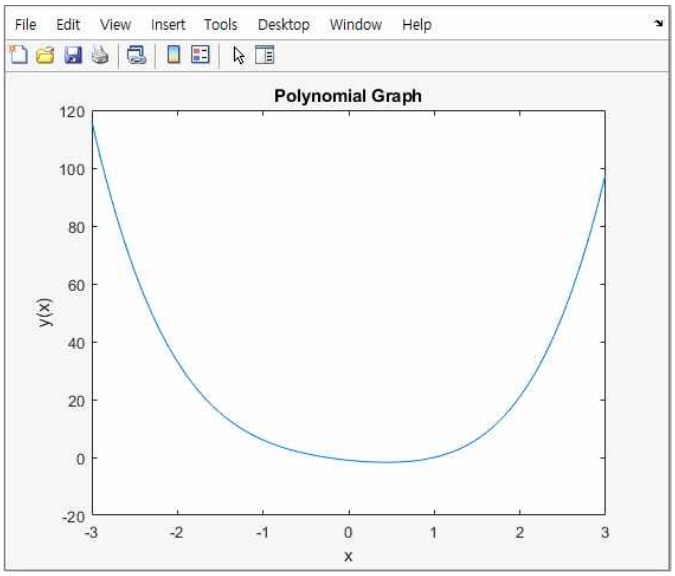
1.12

Command Window

```
>> A = [1 -2 -1 10;-3 -2 4 12;2 -7 0 0];
>> B = [19,13,22,17,23,17,20];
>> C = [24,17,20,22,19,18,16];
>> D = [25,18,22,19,21,19,17];
>> R1 = size(A)
R1 =
     3     4
>> R2 = size(D)
R2 =
     1     7
>> R3 = length(find(B>C&B>D))
R3 =
     2
>> R3 = length(find(B>=D|C<B))
R3 =
     3
```

1.13

```
Command Window
>> x = linspace(-3.0,3.0,61); %or x = -3.0:0.1:3.0;
>> y = x.^4+3*x.^2-3*x-1;
>> plot(x,y);
>> xlabel('x');
>> ylabel('y(x)');
>> title('Polynomial Graph');
```



주석문의 콜론 연산자를 이용하는 배열 **x** = [-3.0:0.1:3.0] 대신에 **linspace** 명령어를 이용하여 입력한다.

1.14

```
Pif_else.m
1 -    grade = input('Enter the score:');
2 -    if(grade >= 50)
3 -        disp("Passed")
4 -    else
5 -        disp("Failed")
6 -    end
```

input 명령어를 이용하여 원하는 문자열을 명령창에 나타낸 후 값을 입력받아 grade에 지정한다.

```
Command Window
>> Pif_else
Enter the score:70
Passed
>> Pif_else
Enter the score:30
Failed
```

1.15

Pelseif.m

```
1 -     grade = input('Enter your score:');
2 -     if(grade >= 90)
3 -         disp("A")
4 -     elseif(grade >= 80)
5 -         disp("B")
6 -     elseif(grade >= 70)
7 -         disp("C")
8 -     elseif(grade >= 60)
9 -         disp("D")
10 -    else
11 -        disp("F")
12 -    end
```

```
Command Window
>> Pelseif
Enter your score:95
A
>> Pelseif
Enter your score:70
C
>> Pelseif
Enter your score:40
F
```

1.16

(a) 최종적으로 num3 = 25 + 5 = 30의 연산을 출력하게 된다.

(b) 최종적으로 num3은 3을 출력하게 된다.

1.17

총합의 상한값을 $N=100$으로 지정한다. for문의 순차적 흐름을 실행하는 색인으로 S를 지정한다. for문을 처음 실행할 때 $k=1$, $S=0$이다.

```
Pfor.m
1 –    N = 100;
2 –    S = 0;
3 –    for k = 1:N
4 –        S = S + 1/k^3;
5 –    end
```

연속적으로 k값을 최종값 $N=100$까지 1씩 증가시켜서 계산한 총합을 나타낸다.

```
Command Window
>> Pfor
>> format long
>> S
S =
   1.202007400659678
```

1.18

총합의 상한값을 $N=100$으로 지정한다. 총합을 나타내는 초기화 값 $S=0$과 총합의 하한값에 대한 초기화 값 $k=1$를 지정한다.

```
Pwhile.m
1 –    N = 100;
2 –    S = 0;
3 –    k = 1;
4 –    while k <= N
5 –        S = S + 1/k^3;
6 –        k = k +1;
7 –    end
```

연속적으로 k값을 최종값 $N=100$까지 1씩 증가시켜서 계산한 총합을 나타낸다.
이것은 [연습문제 1.18]의 for문을 사용한 실행 결과와 동일하다.

```
Command Window
>> Pwhile
>> format long
>> S
S =
   1.202007400659678
```

1.19

(a) b는 최종적으로 5의 값을 출력한다.

(b) b는 최종적으로 4의 값을 출력하고 순환문을 종료한다.

1.20

배열 **pts**에 **length** 명령어를 이용하면 학생 최대 수인 10이 출력된다. 이때 **while**문은 $n<10$의 조건을 만족하는 동안 반복하여 실행한다.

Pwhile_if.m

```
1 -    pts = [84,46,90,92,56,65,73,74,50,40];
2 -    b_avg = 0;
3 -    n = 0;
4 -    while n < length(pts)
5 -        n = n + 1;
6 -        if pts(n) < mean(pts)
7 -            b_avg = b_avg + 1;
8 -        end
9 -    end
10 -   avergae = mean(pts)
11 -   display(b_avg)
```

mean(pts) 명령어를 이용하여 배열 **pts** 의 평균을 구하면 67이고 **if**문에서 **pts** 값이 평균보다 작은 경우만 조건을 만족하는 '참'이므로 **b_avg** 값은 0에서 시작하여 5를 더한 5가 최종값으로 출력된다.

Command Window

```
>> Pwhile_if
avergae =
    67
b_avg =
     5
```

1.21

주어진 행렬 A는 2×3의 크기를 갖고 있으므로 명령어 size(A)를 실행하면 [2,3]이 된다. 결국 $r=2$, $c=3$의 값을 갖게 된다. ones는 행렬의 모든 원소값들을 1로 지정하는 명령어이다.

B는 2행 3열의 6개 원소값 모두를 1로 지정하고 있는 행렬을 표시한다. 프로그램 읽기를 향상시켜 이해도를 높이기 위해서 들여쓰기를 하였는데, 제일 안쪽 내용부터 바깥쪽으로 실행시킨다. 처음 for 순환문의 i값은 1부터 2까지 실행하고 두 번째 for 순환문의 j값은 1부터 3까지 실행한다.

$i=1$과 $j=1$의 값이 if 순환문에 적용되면 A(1,1)<0의 조건을 검사한다. A(1,1)의 의미는 배열 A의 1행 1열 값 1을 말한다. 1은 0보다 큰 값이므로 거짓 조건이 되어 else 순환문을 실행한다. 즉 B(1,1)=A(1,1)을 실행하면 A(1,1)의 값 1을 배열 B의 1행 1열 자리에 지정한다.

명령어 end로 종료되면 다시 안쪽 for 순환문으로 가서 j값을 2로 증가시킨다. 여전히 바깥쪽 for 순환문의 첨자 i값은 1로 유지된다. $i=1$과 $j=2$의 값이 if 순환문에 적용되면 A(1,2)<0의 조건을 검사한다. A(1,2)의 값 3은 0보다 큰 값이므로 거짓 조건이 되어 else 순환문을 실행한다. 즉 B(1,2)=A(1,2)을 실행하면 A(1,2)의 값 3을 배열 B의 1행 2열 자리에 지정한다.

다시 안쪽 for 순환문으로 가서 j값을 3으로 증가시킨다. $i=1$과 $j=3$의 값이 if 순환문에 적용되면 A(1,3)<0의 조건을 검사한다. A(1,3)의 값 −4는 0보다 작은 값이므로 조건을 만족하여 if 순환문을 실행한다. 즉 B(1,3)=B(1,3)을 실행하면 1을 배열 B의 1행 3열 자리에 지정한다.

명령어 end로 종료되면 이제는 바깥쪽 for 순환문으로 가서 i값을 2로 증가시킨다. $i=2$와 $j=1$의 값이 if 순환문에 적용되면 A(2,1)<0의 조건을 검사한다. A(2,1)의 값 −5는 0보다 작은 값이므로 조건을 만족하여 if 순환문을 실행한다. 즉 B(2,1)=B(2,1)을 실행하면 1을 배열 B의 2행 1열 자리에 지정한다.

여전히 바깥쪽 for 순환문의 첨자 i값은 2로 유지된다. $i=2$와 $j=2$의 값이 if 순환문에 적용되면 A(2,2)<0의 조건을 검사한다. A(2,2)의 값 −2는 0보다 작은 값이므로 조건을 만족하여 if 순환문을 실행한다. 즉 B(2,2)=B(2,3) 을 실행하면 1을 배열 B의 2행 2열 자리에 지정한다.

여전히 바깥쪽 for 순환문의 첨자 i값은 2로 유지된다. $i=2$와 $j=3$의 값이 if 순환문에 적용되면 A(2,3)<0의 조건을 검사한다. A(2,3)의 값 4는 0보다 큰 값이므로 거짓 조건이 되어 else 순환문을 실행한다. 즉 B(2,3)=A(2,3)을 실행하면 A(2,3)의 값 4를 배열 B의 2행 3열 자리에 지정한다.

마침내 순환문과 조건문의 모든 실행을 중단하고 다음과 같이 배열 B의 값을 출력하게 된다.

```
Command Window
>> Pfor_if_else
     1     3     1
     1     1     4
```

1.22

```
cir_rhom.m
1  function S = cir_rhom(d)
2  r = d/2;
3  cirarea = (pi*r^2);
4  triarea = 1/2*d*r;
5  rhombus = 2*triarea;
6  S = cirarea - rhombus;
7  end
```

입력변수가 d, 출력변수가 S인 사용자정의함수를 정의한다. 지름 d의 $\frac{1}{2}$인 반지름 r을 지정한다. 변수 cirarea에 원의 면적을 계산하는 식 πr^2을 지정한다. 변수 triarea에 삼각형의 면적, 변수 rhombus에 마름모 면적을 지정한다.

출력변수 S에 구하고자 하는 면적을 계산하는 식 cirarea - rhombus를 지정한다.
선택적으로 function에 상응하는 end 명령어를 지정한다.

```
Command Window
>> S = cir_rhom(10)
S =
   28.5398
```

1.23

paracap.m

```
1   function [C]= paracap(d, r)
2 -   L = 2;
3 -   epsilon_0 = 8.854e-12;
4 -   C = (pi*epsilon_0*L)/log((d-r)/r);
```

function 에 이어서 출력변수는 C를 대괄호를 사용하여 설정한다. 사용자정의함수의 입력변수 d와 r은 소괄호 안에 콤마로 분리하여 설정한다. $L=2$와 $\epsilon_0 = 8.854\times10^{-12}$을 지정하고 주어진 수식을 입력한다.

다음 그림은 먼저 $d=5[\mathrm{mm}]$, $r=2[\mathrm{mm}]$을 지정한 후 사용자정의함수 paracap.m 를 실행한 결과이다.

Command Window

```
>> d = 5e-3
d =
    0.0050
>> r = 2e-3
r =
    0.0020
>> C = paracap(d,r)
C =
   1.3720e-10
```

1.24

factorial.m

```
function F = factorial(n)
    F = 1;
    i = 1;
    while i <= n
        F = F * i;
        i = i + 1;
    end
end
```

입력 변수 n 을 사용한 사용자정의함수 factorial 을 지정한다. while 문을 이용하여 계승 함수의 재귀적 정의를 실행한다.

주어진 3 개의 값에 대한 계승을 계산하면 다음과 같다.

Command Window

```
>> F = factorial(0)
F =
     1
>> F = factorial(1)
F =
     1
>> F = factorial(3)
F =
     6
```

Chapter 02 연습문제

2.1

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

2.2

Command Window

```
>> A = [4 -2 -1;-1 2 -1;-1 -2 3];
>> c = [0;3;3];
>> x = A₩c
x =
    9.0000
   12.0000
   12.0000
```

2.3

Command Window

```
>> A = [4 -2 -1;-1 2 -1;-1 -2 3];
>> c = [0;3;3];
>> x = inv(A)*c
x =
    9.0000
   12.0000
   12.0000
```

2.4

```
Command Window
>> A = [4 -2 -1;-1 2 -1;-1 -2 3];
>> c = [0;3;3];
>> D = det(A);
>> A1 = [c(:,1),A(:,2),A(:,3)];
>> A2 = [A(:,1),c(:,1),A(:,3)];
>> A3 = [A(:,1),A(:,2),c(:,1)];
>> x1 = det(A1)/D
x1 =
    9.0000
>> x2 = det(A2)/D
x2 =
   12.0000
>> x3 = det(A3)/D
x3 =
   12.0000
```

2.5

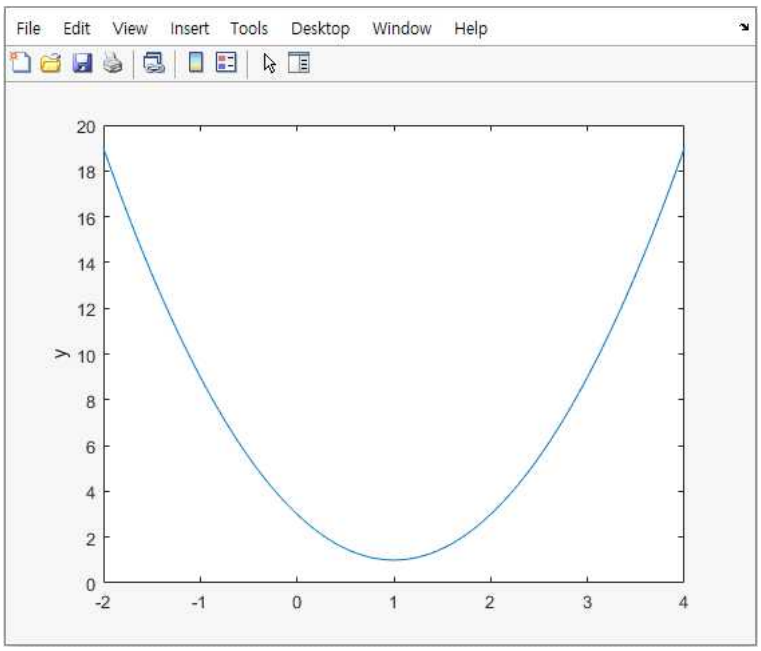
2차 다항식에 세 개의 좌표점을 대입하면 다음과 같은 세 개의 선형연립방정식을 형성한다. 여기에 MATLAB의 왼쪽 나눗셈을 이용하여 미지수 p, q, r을 구하면 된다.

```
Command Window
>> A = [1,1,1;4,-2,1;16,-4,1];
>> c = [1;19;51];
>> x = A\c;
>> p = x(1)
p =
     2
>> q = x(2)
q =
    -4
>> r = x(3)
r =
     3
```

2.6

계수값을 구하고 찾아낸 2 차 곡선은 $y=2x^2-4x+3$이다. polyval 함수를 이용하면 그래프를 나타낼 수 있다.

```
Command Window
>> x = [-2:0.01:4];
>> c = [2,-4,3];
>> y = polyval(c,x);
>> plot(x,y);
>> xlabel('x');
>> ylabel('y');
```



2.7.

사용자정의함수는 다음과 같다.

PF2_07.m

```
function y = PF2_07(x)
y = exp(-0.2*x).*cos(x+2)+0.1;
end
```

주어진 구간에서 세 개의 근들은 구하면 각각 2.5452, 6.2074, 8.4259이다.

Command Window

```
>> fzero('PF2_07',2)
ans =
    2.5452
>> fzero('PF2_07',5)
ans =
    6.2074
>> fzero('PF2_07',8)
ans =
    8.4259
```

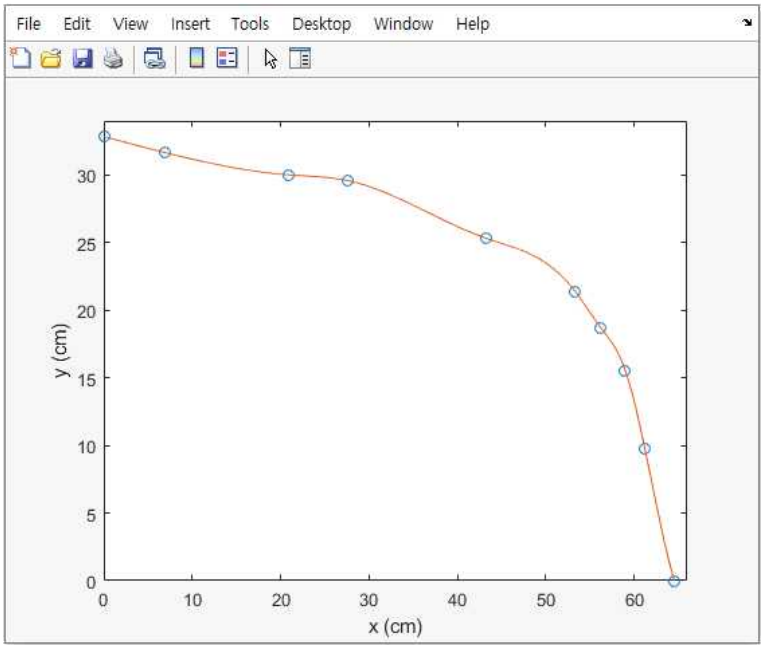
2.8

다음 그림에 스크립트 파일을 보여주고 있다.

car_spline.m

```
x = [0,6.92,20.94,27.62,43.26,53.34,56.22,59.0,61.23,64.55];
y = [32.84,31.66,30.01,29.58,25.32,21.39,18.72,15.56,9.74,0];
x_int = 0:0.01:65;
y_int = spline(x,y,x_int);
plot(x,y,'o',x_int,y_int);
xlabel('x (cm)');
ylabel('y (cm)');
axis([0 66 0 34]);
```

다음 그림은 스크립트 파일을 실행하여 생긴 곡선 그래프를 보여주고 있다. 작은 원들은 표에 주어진 각각의 좌표들의 위치를 표시한다. 이들 점에 대하여 spline 함수를 실행한 결과로 작은 원들을 통과하는 곡선이 그려져 있다.



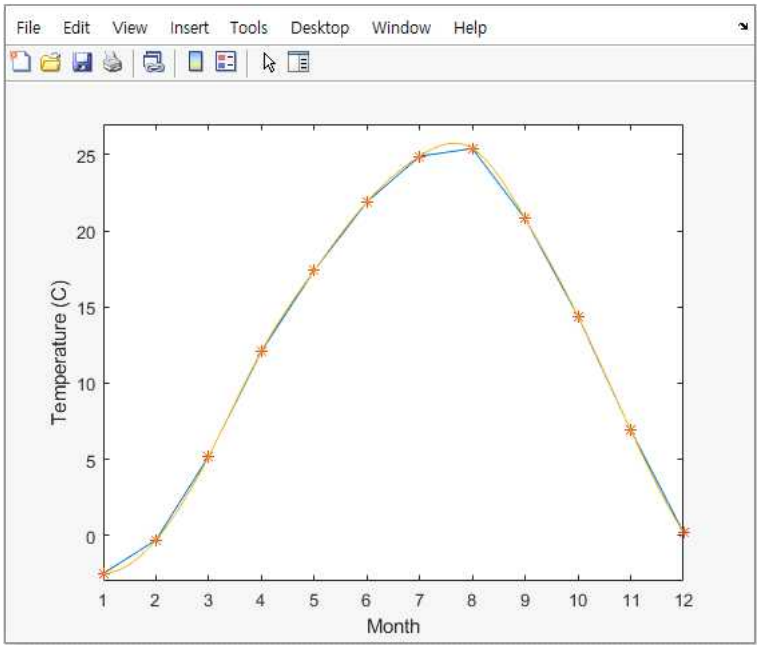
2.9

다음 그림은 스크립트 파일을 보여주고 있다. 배열 연산자를 이용하여 1월부터 12월을 배열 m으로 표시하였고 각 달의 평균 기온을 배열 T로 표시하였다. 대략적으로 하루는 0.03의 숫자로 표시하여 m_int에 지정하였다.

temp_spline.m

```
1 -    m = 1:12;
2 -    T = [-2.5,-0.3,5.2,12.1,17.4,21.9,24.9,25.4,20.8,14.4,6.9,0.2];
3 -    m_int = 1:0.03:12;
4 -    T_int = spline(m,T,m_int);
5 -    plot(m,T,m,T,'*',m_int,T_int);
6 -    xlabel('Month');
7 -    ylabel('Temperature (C)');
8 -    axis([1 12 -3 27]);
```

다음 그림은 스크립트 파일을 실행하여 나타난 그래프이다.



다음 그림은 4월 5일, 6월 6일, 8월 15일, 11월 20일의 기온을 추정한 결과들을 보여주고 있다. 추정하려는 날짜의 숫자 변환은 1을 각 달의 전체 일수로 나누고 다시 해당 수를 곱해서 나타내었다. 예를 들면 4월 달은 30이 전체 일수이므로 1에서 30으로 나누고 다시 5일에 해당하는 5를 곱한 값인 0.17를 4에 더한 4.17를 추정값으로 변환하였다.

```
Command Window
>> m = 1:12;
>> T = [-2.5,-0.3,5.2,12.1,17.4,21.9,24.9,25.4,20.8,14.4,6.9,0.2];
>> m_est = [4.17 7.20 8.48 11.67];
>> T_est_L = interp1(m,T,m_est)
T_est_L =
   13.0010   25.0000   23.1920    2.4110
>> T_est_S = interp1(m,T,m_est,'spline')
T_est_S =
   13.1259   25.3006   23.6740    2.1668
```

2.10

MATLAB의 함수 **trapz**를 사용하기 이전에 수치적으로 수영장 내부의 높이 h와 입방체 흐름 속도 f의 관계를 수식으로 표현하면 $10\frac{dh}{dt}=f$가 되는데, 이것을 적분식으로 다시 변환하면 $h=0.1\int_0^t f dt$가 된다. 이 적분식에 대해서 함수 **trapz**를 실행한 결과를 다음 그림에서 보여주고 있다. 폭우가 쏟아진 14분후 수영장 내부의 물 높이는 4.6635 m를 표시하게 된다.

```
Command Window
>> t = 0:14;
>> f = [0,2.37,3.80,4.47,4.39,4.62,4.98,4.63,3.89,3.29,3,2.7,2.45,1.57,0.95];
>> h = 0.1*trapz(t,f)
h =
    4.6635
```

2.11

syms 명령어로 기본 독립변수 t의 인자를 갖는 함수 $x(t)$와 $y(t)$를 지정한다. **diff** 함수를 이용하여 x와 y에 대한 1차 미분방정식 Dx와 Dy로 지정한다. **dsolve** 함수를 이용하여 1차 미분방정식을 실행하면 배열 형태의 결과를 얻는다.

```
Command Window
>> syms x(t) y(t);
>> Dx = diff(x);
>> Dy = diff(y);
>> [x,y] = dsolve(Dx==4*x+3*y,Dy==-3*x+4*y)
x =
C2*cos(3*t)*exp(4*t) + C1*sin(3*t)*exp(4*t)
y =
C1*cos(3*t)*exp(4*t) - C2*sin(3*t)*exp(4*t)
```

2.12

```
Command Window
>> syms x(t) y(t);
>> Dx = diff(x);
>> Dy = diff(y);
>> [x,y] = dsolve(Dx==4*x+3*y,Dy==-3*x+4*y,x(0)==1,y(0)==0)
x =
cos(3*t)*exp(4*t)
y =
-sin(3*t)*exp(4*t)
```

syms 명령어로 기본 독립변수 t의 인자를 갖는 함수 $x(t)$와 $y(t)$를 지정한다. diff 함수를 이용하여 x와 y에 대한 1차 미분방정식 Dx와 Dy로 지정한다. dsolve 함수에 두 개 초깃값을 이용하여 주어진 연립 1차 미분방정식을 실행하면 배열 형태의 결과를 얻는다.

2.13

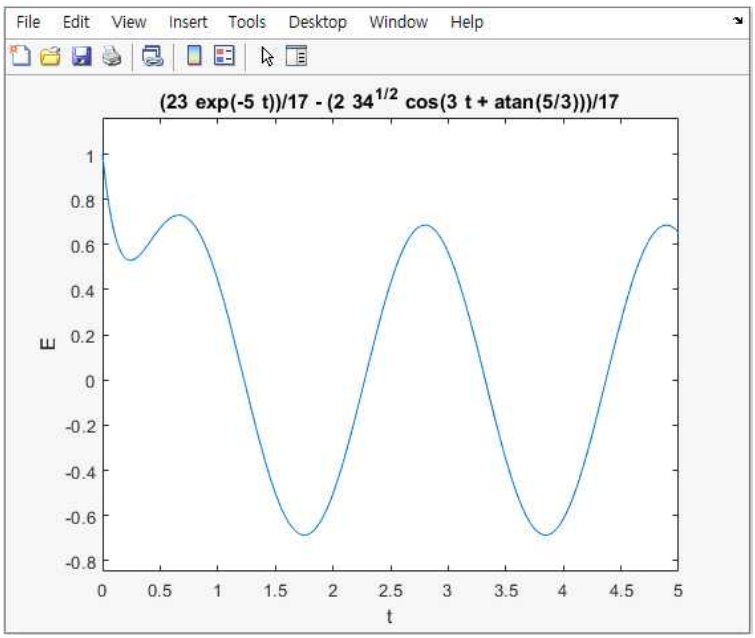
```
Command Window
>> syms y(t);
>> Dy = diff(y);
>> D2y = diff(y,2);
>> D3y = diff(y,3);
>> y = dsolve(D3y-5*Dy+10==1,y(0)==1,Dy(0)==0,D2y(0)==0)
y =
(9*t)/5 - (9*5^(1/2)*exp(5^(1/2)*t))/50 + (9*5^(1/2)*exp(-5^(1/2)*t))/50 + 1
```

syms 명령어를 이용하여 기본 독립변수 t의 인자를 갖는 함수 $y(t)$를 지정한다. diff 함수를 이용하여 y를 실행한 값을 1차, 2차, 3차 미분방정식 Dy, $D2y$, $D3y$로 지정한다. dsolve 함수에 3개 초깃값을 이용하여 주어진 3차 미분방정식을 각각 실행한다.

2.14

```
Command Window
>> syms y(t);
>> Dy = diff(y);
>> E = dsolve(Dy==4*sin(3*t)-5*y,y(0)==1);
>> ezplot(E,0,5);
>> ylabel('E');
```

syms 명령어를 이용하여 기본 독립변수 t의 인자를 갖는 함수 $y(t)$와 계수를 지정한다. diff 함수를 이용하여 y를 실행한 값을 1차 미분방정식 Dy로 지정한다. dsolve 함수를 이용하여 주어진 1차 미분방정식과 초깃값을 입력한 후 실행한다. ezplot 함수를 이용하여 1차 미분방정식을 그래프로 나타낸다.



Chapter 03 연습문제

3.1

```
Command Window
>> bin2dec('101011')
ans =
    43
>> dec2bin(43)
ans =
    '101011'
```

3.2

4비트씩 사용한 부분에 대한 10진수 값을 조합하면 16진수는 다음과 같이 $1C_{16}$이 된다.

$$\underbrace{0\ \ 0\ \ 0\ \ 1}_{1}\ \underbrace{1\ \ 1\ \ 0\ \ 0}_{C}$$

3.3

```
Command Window
>> dec2base(28,8)
ans =
    '34'
>> dec2base('28',8)
ans =
  2×2 char array
    '62'
    '70'
>> base2dec('34',8)
ans =
    28
>> base2dec(34,8)
Error using base2dec>base2decImpl (line 58)
'"' contains characters which cannot be converted to base 8.
Error in base2dec (line 26)
        dec = base2decImpl(str,base);
```

3.4

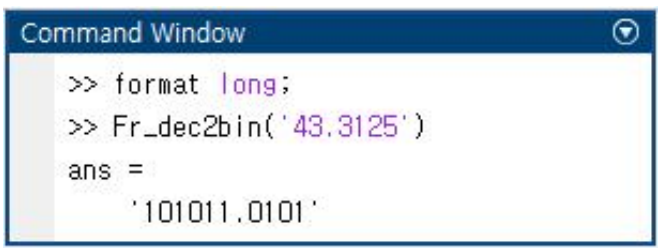
10진 정수 43을 2진 정수로 변환하려면 다음과 같은 계산 과정을 실행한다.

$$
\begin{array}{lll}
2\overline{)43} = 21 & & b_0 = 1 \\
\quad 2\overline{)21} = 10 & & b_1 = 1 \\
\quad\quad 2\overline{)10} = 5 & & b_2 = 0 \\
\quad\quad\quad 2\overline{)5} = 2 & & b_3 = 1 \\
\quad\quad\quad\quad 2\overline{)2} = 1 & & b_4 = 0 \\
\quad\quad\quad\quad 2\overline{)1} = 0 & & b_5 = 1
\end{array}
$$

10진 부동 소수점 수 0.3125를 2진 부동 소수점 수로 변환하려면

$$
\begin{aligned}
&D = D_1 = 0.3125 \\
&2D_1 = 0.625 = D_2 \rightarrow b_1 = 0 \\
&2D_2 = 0.25 = D_3 \rightarrow b_2 = 1 \\
&2D_3 = 0.5 = D_4 \rightarrow b_3 = 0 \\
&2D_4 = 0.0 = D_5 \rightarrow b_4 = 1
\end{aligned}
$$

2진 정수와 2진 부동 소수점 수를 같이 붙여 쓴 $(101011.0101)_2$이 최종 변환된 값을 나타낸다.



3.5

$(10.0011)_2 = 2^1 + 2^{-3} + 2^{-4} = 2.1875$

$(1.11101)_2 = 2^0 + 2^{-1} + 2^{-2} + 2^{-3} + 2^{-5} = 1.90625$

두 수를 더하면 다음과 같은 10 진 부동 소수점 수를 얻는다.
$2.1875 + 1.90625 = 4.09375$

```
Command Window
>> format long;
>> D1 = Fr_bin2dec('10.0011');
>> D2 = Fr_bin2dec('1.11101');
>> D = D1+D2
D =
   4.093750000000000
```

3.6

주어진 2 진 소수 $(1.0\overline{1001})_2$는 1001이 계속 반복되는 순환소수를 나타내고 있다.
여기에 식 (3.6)을 이용하면 공비는 $|r| = |2^{-4}| < 1$가 되어 10 진 소수는 다음과 같다.

$$\left(\frac{1}{4} + \frac{1}{32}\right)\left(\frac{1}{1-1/16}\right) = \frac{9}{32} \times \frac{16}{15} = \frac{3}{10} = 0.3$$

$$(1.0\overline{1001})_2 = 1.3$$

3.7

(1) 잘라버리기

왼쪽 2진수 111_2에 오른쪽 방향으로 나열되어 있는 2진수를 연속으로 더하는 문제이다. 가장 중요한 과정은 지수승의 최댓값을 먼저 고려하고 동시에 소수점 세 자리를 맞추어야만 한다.

그러므로 최종 결과는 다음과 같다.
$111_2 + 11_2 + 1_2 + 0.1_2 = (1.011)_2 \cdot 2^3 + (0.000)_2 \cdot 2^3 = (1.011)_2 \cdot 2^3$

(2) 끝수처리

잘라버림과 동일한 조건으로 끝수처리 계산도 실행하게 된다. 그러나 더하는 2진수 0.1_2을 지수승 3에 맞추면 $0.0001_2 \cdot 2^3$이 되고 소수점 네 자리에서 반올림이 적용되어 $0.001_2 \cdot 2^3$로 표시된다. 계산 과정은 다음과 같다.

$$111_2 + 11_2 + 1_2 + 0.1_2 = (1.011)_2 \cdot 2^3 + (0.001)_2 \cdot 2^3 = (1.100)_2 \cdot 2^3$$

3.8

(a) 3비트 저장 기능만 있는 2진 컴퓨터이기 때문에 본문의 식 (3.8) $x = \sigma \cdot \overline{x} \cdot 2^e$의 조건에 맞추어서 문제를 풀어야 한다.

$$5+4+3+2+1 = (110)_2 \cdot 2^1 + (000)_2 \cdot 2^1 = (110)_2 \cdot 2^1 = 12$$

(b) 반면에 주어진 덧셈을 오른쪽에서 왼쪽으로 실행하면 먼저 제일 오른쪽 1을 왼쪽 2에 더하면

$$1+2+3+4+5 = (101)_2 \cdot 2^1 + (010)_2 \cdot 2^1 = (111)_2 \cdot 2^1 = 14$$

3.9

이 문제는 컴퓨터의 저장 장치의 정밀도가 얼마나 중요한가를 잘 보여 주는 보기이다. 1을 계속해서 50차례 더해가는 계산인데, 왼쪽에서 오른쪽 혹은 오른쪽에서 왼쪽 중 어느 방향으로 실행을 하더라도 그 결과는 다음과 같이 동일하게 된다.

최종 실행 결과는 다음과 같다.

$$1+1+1+1+1+\cdots+1 = (10)_2 \cdot 2^1 + 0 + 0 + \cdots + 0 = 4$$

3.10

(a) 근삿값 x_1의 경우는 최소 지정 자리수가 주어진 조건보다 작은 값을 갖게 되어 언더 플로우 오류가 발생되어 근삿값은 $\hat{x}_1 = 0$이 된다. 근삿값 x_2는 유효숫자가 다섯 자리이므로 정수 한 자리와 부동 소수점 수 네 자리만 표시하므로 부동 소수점 수 다섯 자리에서 올림이 적용되어 $\hat{x}_2 = 6.0222 \times 10^{21}$가 된다.

(b) 주어진 값 x_2에 대한 절대 오차와 상대 오차는 다음과 같이 계산된다.

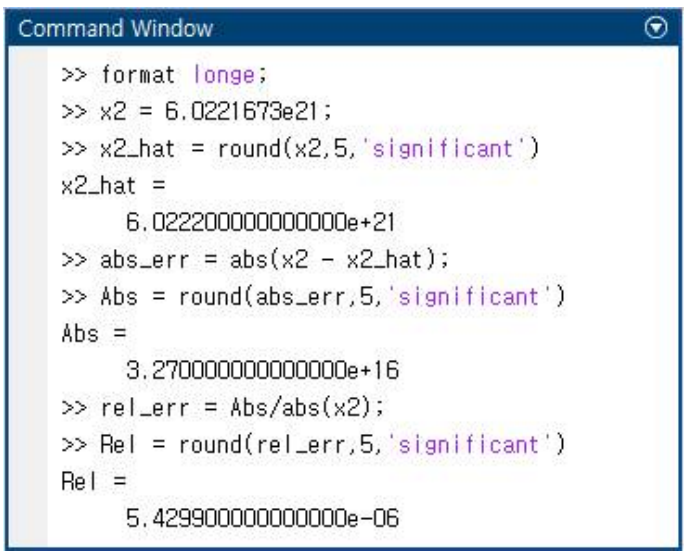
$$\left|x_2 - \hat{x}_2\right| = \left|6.0221673 \times 10^{21} - 6.0222 \times 10^{21}\right| = 0.0000327 \times 10^{21}$$

$$\left|x_2 - \hat{x}_2\right| = 3.2700 \times 10^{16}$$

$$\left|\frac{x_2 - \hat{x}_2}{x_2}\right| = \left|\frac{3.2700 \times 10^{16}}{6.0221673 \times 10^{21}}\right| = 5.2499 \times 10^{-6}$$

3.11

다음 그림과 같이 MATLAB 의 **round** 함수를 이용하여 x_2에 유효숫자 다섯 자리를 지정하여 올림을 실행한다. 절대 오차를 계산한 뒤 다시 round 함수를 이용하면 유효숫자 다섯 자리를 나타내는 절대 오차를 계산할 수 있다. 같은 방법으로 유효숫자 다섯 자리를 나타내는 상대 오차를 계산한다.



3.12

식 (1)을 이용하면

$$x_{1,2} = \frac{-(-15) \pm \sqrt{(-15)^2 - 4(1)(1)}}{2(1)} = \frac{15 \pm 14.9}{2}$$

$x_1 = 14.95$과 $x_2 = 0.0500$이다. 식 (2)를 이용하면

$$x_{1,2} = \frac{-2(1)}{(-15) \pm \sqrt{(-15)^2 - 4(1)(1)}} = \frac{-2}{-15 \pm 14.9}$$

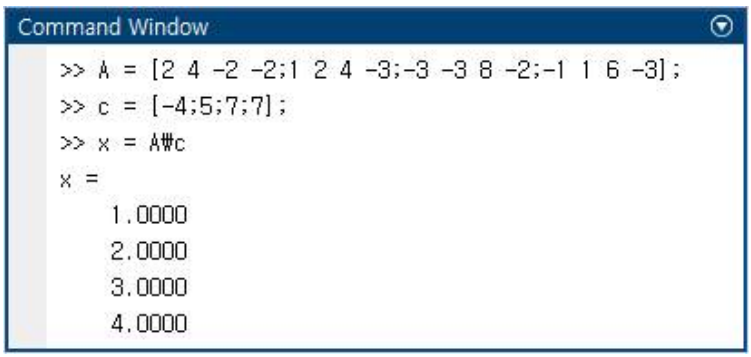
$x_1 = 20.0$과 $x_2 = 0.0669$이다.

실제값은 $x_1 = 14.93$과 $x_2 = 0.0670$이다. x_1의 경우 식 (1)의 x_1이 식 (2)의 x_1보다 오차가 작지만, x_2의 경우 식 (2)의 x_2가 식 (1)의 x_2보다 오차가 작다. 이처럼 식 (1)과 식 (2)로 구한 두 쌍의 근의 오차가 서로 다름을 보여주고 있다.

Chapter 04 연습문제

4.1

$$A = \begin{bmatrix} 2 & 4 & -2 & -2 \\ -3 & -3 & 8 & -2 \\ 1 & 2 & 4 & -3 \\ -1 & 1 & 6 & -3 \end{bmatrix}, \quad c = \begin{bmatrix} -4 \\ 7 \\ 5 \\ 7 \end{bmatrix}$$



4.2

$$A-B = \begin{bmatrix} 2-(1) & 4-(0) & -3-(1) \\ 1-(0) & 6-(1) & 1-(-1) \\ -1-(2) & -2-(3) & -3-(3) \end{bmatrix} = \begin{bmatrix} 1 & 4 & -4 \\ 1 & 5 & 2 \\ -3 & -5 & -6 \end{bmatrix}$$

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

행렬 C의 행과 열의 원소들을 서로 바꾸면 다음과 같다.

$$C^T = \begin{bmatrix} 0 & 1 & -3 \\ 4 & 4 & -5 \\ -4 & 2 & -7 \end{bmatrix}$$

MATLAB 실행 과정을 그림에서 보여주고 있다. eye(3) 명령어를 이용하여 3행 3열 단위행렬을 간단하게 지정한다. 아포스트로피를 이용하여 행렬 C의 전치행렬을 실행한다.

```
Command Window
>> A = [2 4 -3;1 6 1;-1 -2 -3];
>> B = [1 0 1;0 1 -1;2 3 3];
>> I = eye(3);
>> C = A - B - I;
>> C_tran = C'
C_tran =
     0     1    -3
     4     4    -5
    -4     2    -7
```

4.3

그림에서 보여주듯이 덧셈에 대한 결합 법칙이 성립하고 있다.

```
Command Window
>> A = [2 4 -3;1 6 1;-1 -2 -3];
>> B = [1 0 1;0 1 -1;2 3 3];
>> I = eye(3);
>> C_tran = (A-B-I)';
>> (A+B)+C_tran
ans =
     3     5    -5
     5    11    -5
    -3     3    -7
>> A+(B+C_tran)
ans =
     3     5    -5
     5    11    -5
    -3     3    -7
```

4.4

곱셈에 대한 교환 법칙 AB = BA는 성립 되지 않는다.

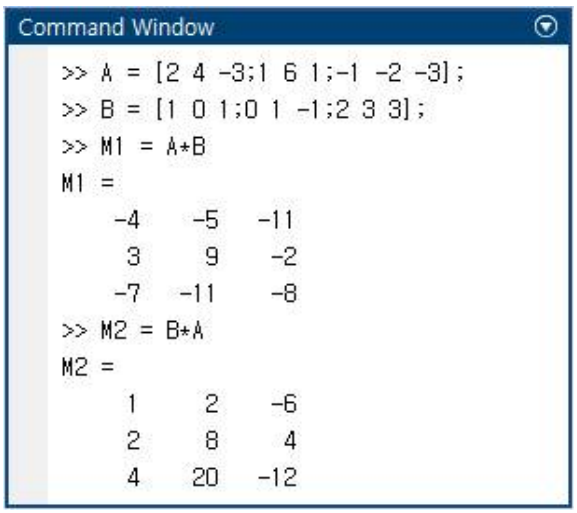
$$AB = \begin{bmatrix} (2)(1)+0+(-3)(2) & 0+(4)(1)+(-3)(3) & (2)(1)+(4)(-1)+(-3)(3) \\ (1)(1)+0+(1)(2) & 0+(6)(1)+(1)(3) & (1)(1)+(6)(-1)+(1)(3) \\ (-1)(1)+0+(-3)(2) & 0+(-2)(1)+(-3)(3) & (-1)(1)+(-2)(-1)+(-3)(3) \end{bmatrix}$$

$$= \begin{bmatrix} -4 & -5 & -11 \\ 3 & 9 & -2 \\ -7 & -11 & -8 \end{bmatrix}$$

$$BA = \begin{bmatrix} (1)(2)+0+(1)(-1) & (1)(4)+0+(1)(-2) & (1)(-3)+0+(1)(-3) \\ 0+(1)(1)+(-1)(-1) & 0+(1)(6)+(-1)(-2) & 0+(1)(1)+(-1)(-3) \\ (2)(2)+(3)(1)+(3)(-1) & (2)(4)+(3)(6)+(3)(-2) & (2)(-3)+(3)(1)+(3)(-3) \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 2 & -6 \\ 2 & 8 & 4 \\ 4 & 20 & -12 \end{bmatrix}$$

MATLAB의 실행 결과는 다음과 같다.



4.5

곱셈에 대한 배분 법칙이 성립 되는 것을 다음 그림에서 보여 주고 있다.

```
Command Window
>> A = [-7 -5 2;10 6 1;3 -9 8];
>> B = [3 -2 1;6 8 -5;7 9 10];
>> C = [6 9 -4;7 5 3;-8 2 1];
>> M = A*(B+C)
M =
  -130   -92    53
   167   159   -31
   -98    -8    97
>> D1 = A*B;
>> D2 = A*C;
>> D = D1+D2
D =
  -130   -92    53
   167   159   -31
   -98    -8    97
```

4.6

행렬식이 $\det(\mathbf{A})=5$인 정칙행렬이므로 역행렬의 연산이 가능하다.

$$\mathbf{A}^{-1}=\frac{1}{5}\begin{bmatrix}1 & 8 & 5\\ 0 & -10 & -5\\ 1 & -7 & -5\end{bmatrix}=\begin{bmatrix}0.2 & 1.6 & 1\\ 0 & -2 & -1\\ 0.2 & -1.4 & -1\end{bmatrix}$$

```
Command Window
>> A = [3 1 2;-1 -2 1;2 3 -2];
>> det(A)
ans =
    5.0000
>> inv(A)
ans =
    0.2000    1.6000    1.0000
   -0.0000   -2.0000   -1.0000
    0.2000   -1.4000   -1.0000
```

4.7

선형시스템의 각각 배열 형태는 다음과 같이 나타낸다.

$$\mathbf{A}=\begin{bmatrix}3 & 1 & 2\\ -1 & -2 & 1\\ 2 & 3 & -2\end{bmatrix},\qquad \mathbf{x}=\begin{bmatrix}x_1\\ x_2\\ x_3\end{bmatrix},\qquad \mathbf{c}=\begin{bmatrix}6\\ 3\\ -5\end{bmatrix}$$

[연습문제 4.6]의 행렬 $\mathbf{A}$와 동일하므로 [연습문제 4.6]에서 구한 역행렬 $\mathbf{A}^{-1}$과 상수벡터 $\mathbf{c}$를 이용하면 변수 벡터 $\mathbf{x}=\mathbf{A}^{-1}\mathbf{c}$를 계산하면 다음과 같다.

$$\mathbf{A}^{-1}=\frac{1}{5}\begin{bmatrix}1 & 8 & 5\\ 0 & -10 & -5\\ 1 & -7 & -5\end{bmatrix}=\begin{bmatrix}0.2 & 1.6 & 1\\ 0 & -2 & -1\\ 0.2 & -1.4 & -1\end{bmatrix}$$

$$\begin{bmatrix}x_1\\ x_2\\ x_3\end{bmatrix}=\begin{bmatrix}0.2 & 1.6 & 1\\ 0 & -2 & -1\\ 0.2 & -1.4 & -1\end{bmatrix}\begin{bmatrix}6\\ 3\\ -5\end{bmatrix}=\begin{bmatrix}(0.2)(6)+(1.6)(3)+(1)(-5)\\ (0)(6)+(-2)(3)+(-1)(-5)\\ (0.2)(6)+(-1.4)(3)+(-1)(-5)\end{bmatrix}\begin{bmatrix}1\\ -1\\ 2\end{bmatrix}$$

4.8

본문의 [그림4-5]를 기초로 세 개 선형방정식은 계수행렬 A는 3행3열, 상수벡터 c는 3행1열로 확장된다. 상수벡터를 계수행렬에 대치하는 행렬의 행렬식은 3개가 되고 미지의 변수도 3개가 된다.

Pcramer.m

```
A = input('Enter a 3-by-3 matrix A: ');
c = input('Enter a 3-by-1 matrix c: ');
D = det(A);
A1 = [c(:,1),A(:,2),A(:,3)];
A2 = [A(:,1),c(:,1),A(:,3)];
A3 = [A(:,1),A(:,2),c(:,1)];
x1 = det(A1)/D;
x2 = det(A2)/D;
x3 = det(A3)/D;
disp('x1 = ');
disp(x1)
disp('x2 = ');
disp(x2)
disp('x3 = ');
disp(x3)
```

4.9

```
Command Window
>> Pcramer
Enter a 3-by-3 matrix A: [3 1 2;-1 -2 1;2 3 -2]
Enter a 3-by-1 matrix c: [6;3;-5]
x1 =
    1.0000
x2 =
   -1.0000
x3 =
     2
```

4.10

확대행렬은 다음과 같다.

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

후진 대입법을 이용하여 차례대로 변수들을 풀면 $x_3 = 2,\ x_2 = -1,\ x_1 = 1$이다.

4.11

MATLAB의 사용자정의함수 Gauss를 이용하여 얻은 동일한 변수들의 값들이 그림에서 보여주고 있다.

```
Command Window
>> A = [1 5 -1 6;2 -1 1 -2;-1 4 -1 3;3 -7 -2 1];
>> c = [19;7;20;-75];
>> Gauss(A,c)
ans =
    2.0000
   10.0000
    3.0000
   -5.0000
```

4.12

먼저 확대행렬을 써보자

$$\left[\begin{array}{cccc|c} 2 & 4 & -2 & -2 & -4 \\ 1 & 2 & 4 & -3 & 5 \\ -3 & -3 & 8 & -2 & 7 \\ -1 & 1 & 6 & -3 & 7 \end{array}\right]$$

후진 대입법을 이용하여 차례대로 변수들을 풀면 $x_4 = 4,\ x_3 = 3,\ x_2 = 2,\ x_1 = 1$이다.

4.13

단위행렬과 계수행렬을 묶은 확대행렬은 다음과 같다.

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

$$\mathrm{A}^{-1} = \begin{bmatrix} -0.7 & 0.2 & 0.3 \\ -1.3 & -0.2 & 0.7 \\ 0.8 & 0.2 & -0.2 \end{bmatrix}$$

4.14

앞에서 구한 동일한 확대행렬을 이용한다.

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

[연습문제 4.13]의 역행렬과 동일한 결과를 얻게 된다.

$$\mathrm{A}^{-1} = \begin{bmatrix} -0.7 & 0.2 & 0.3 \\ -1.3 & -0.2 & 0.7 \\ 0.8 & 0.2 & -0.2 \end{bmatrix}$$

4.15

$$m_{21}=\frac{a_{21}^{(1)}}{a_{11}^{(1)}}=-\frac{3}{2},\qquad m_{31}=\frac{a_{31}^{(1)}}{a_{11}^{(1)}}=\frac{1}{2},\qquad m_{41}=\frac{a_{41}^{(1)}}{a_{11}^{(1)}}=-\frac{1}{2}$$

$$m_{32}=\frac{a_{32}^{(2)}}{a_{22}^{(2)}}=0,\qquad m_{42}=\frac{a_{42}^{(2)}}{a_{22}^{(2)}}=1$$

$$m_{43}=\frac{a_{43}^{(3)}}{a_{33}^{(3)}}=0$$

식 (4.21)을 4행4열 행렬로 확장시켜서 이용하면 행렬 L과 행렬 U는 다음과 같다.

$$\mathrm{L}=\begin{bmatrix}1 & 0 & 0 & 0\\ -3/2 & 1 & 0 & 0\\ 1/2 & 0 & 1 & 0\\ -1/2 & 1 & 0 & 1\end{bmatrix},\qquad \mathrm{U}=\begin{bmatrix}2 & 4 & -2 & -2\\ 0 & 3 & 5 & -5\\ 0 & 0 & 5 & -2\\ 0 & 0 & 0 & 1\end{bmatrix}$$

변수 벡터 y의 값들은 다음과 같다.

$$\begin{bmatrix}y_1\\ y_2\\ y_3\\ y_4\end{bmatrix}=\begin{bmatrix}-4\\ 1\\ 7\\ 4\end{bmatrix}$$

Ux = y의 변수 벡터 x를 후진 대입법을 이용하여 구한다.

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

상하행렬 분해를 이용하여 구한 변수 벡터 x는 다음과 같다.

$$\begin{bmatrix}x_1\\ x_2\\ x_3\\ x_4\end{bmatrix}=\begin{bmatrix}1\\ 2\\ 3\\ 4\end{bmatrix}$$

4.16

행렬 U의 1행은 다음과 같다.

$$u_{11} = a_{11} = 3, \quad u_{12} = a_{12} = 1, \quad u_{13} = a_{13} = 2$$

행렬 L과 행렬 U의 나머지 원소는 다음과 같이 구한다.

$$l_{21} = \frac{a_{21}}{u_{11}} = -\frac{1}{3}, \quad l_{31} = \frac{a_{31}}{u_{11}} = \frac{2}{3}$$

$$u_{22} = a_{22} - l_{21}u_{12} = -2 - \left(-\frac{1}{3}\right)(1) = -\frac{5}{3}$$

$$u_{23} = a_{23} - l_{21}u_{13} = 1 - \left(-\frac{1}{3}\right)(2) = \frac{5}{3}$$

$$l_{32} = \frac{a_{32} - l_{31}u_{12}}{u_{22}} = \frac{3 - \left(\frac{2}{3}\right)(1)}{-\frac{5}{3}} = -\frac{7}{5}$$

$$u_{33} = a_{33} - l_{31}u_{13} - l_{32}u_{23} = -2 - \left(\frac{2}{3}\right)(2) - \left(-\frac{7}{5}\right)\left(\frac{5}{3}\right) = -1$$

행렬 L과 U는 다음과 같다.

$$\mathbf{L} = \begin{bmatrix} 1 & 0 & 0 \\ -1/3 & 1 & 0 \\ 2/3 & -7/5 & 1 \end{bmatrix}, \qquad \mathbf{U} = \begin{bmatrix} 3 & 1 & 2 \\ 0 & -5/3 & 5/3 \\ 0 & 0 & -1 \end{bmatrix}$$

변수 벡터 y의 값은 다음과 같다.

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} 6 \\ 5 \\ -2 \end{bmatrix}$$

y값을 적용하면, 다음과 같이 $\mathbf{U}\mathbf{x} = \mathbf{y}$의 형태로 나타낼 수 있다.

$$\begin{bmatrix} 3 & 1 & 2 \\ 0 & -5/3 & 5/3 \\ 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 6 \\ 5 \\ -2 \end{bmatrix}$$

첫 번째 변수는 다음과 같다.

$$x_3 = \frac{y_3}{u_{33}} = \frac{-2}{-1} = 2$$

식 (A4.6)을 이용하여 나머지 변수들을 구하는 과정은 다음과 같다.

$$x_2 = \frac{y_2 - u_{23}x_3}{u_{22}} = \frac{5 - \left(\frac{5}{3}\right)(2)}{-\frac{5}{3}} = -1$$

$$x_1 = \frac{y_1 - u_{12}x_2 - u_{13}x_3}{u_{11}} = \frac{6 - (1)(-1) - (2)(2)}{3} = 1$$

상하행렬 분해를 이용하여 구한 변수 벡터 $\mathbf{x}$는 다음과 같다.

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

4.17

$$\mathrm{L} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 2/3 & 1 & 0 & 0 \\ -1/3 & 5/22 & 1 & 0 \\ 1/3 & 1/2 & -7/11 & 1 \end{bmatrix}, \quad \mathrm{U} = \begin{bmatrix} 3 & -7 & -2 & 1 \\ 0 & 22/3 & -1/3 & 17/3 \\ 0 & 0 & 15/6 & -33/6 \\ 0 & 0 & 0 & -16/11 \end{bmatrix}$$

행렬 L은 자리바꿈을 고려하여 다시 정렬시킨다. 첫 번째 자리바꿈에서 1 행과 4 행을 바꿨지만 하삼각행렬의 기본 형식에 의해서 1 행 1 열의 원소는 항상 1이므로 1 행 4 열의 원소 $\frac{1}{3}$과는 자리바꿈은 불가능하다. 두 번째 자리바꿈에서 2 행과 4 행을 자리바꿈했기 때문에 승수 m_{21}과 m_{41}을 바꾸면 다음과 같다.

$$\mathrm{L} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1/3 & 1 & 0 & 0 \\ -1/3 & 5/22 & 1 & 0 \\ 2/3 & 1/2 & -7/11 & 1 \end{bmatrix}$$

세 번째 자리바꿈에서 3행과 4행을 자리바꿈했기 때문에 승수 m_{32}와 m_{42} 그리고 m_{31}와 m_{41}를 자리바꿈 하면 최종적으로 구하는 행렬 L은 다음과 같다.

$$\mathtt{L} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1/3 & 1 & 0 & 0 \\ 2/3 & 1/2 & 1 & 0 \\ -1/3 & 5/22 & -7/11 & 1 \end{bmatrix}$$

자리바꿈을 적용시킨 새로운 상수 벡터 $\mathtt{d}$는 다음과 같다.

$$\mathtt{d} = \mathtt{Pc} = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} 19 \\ 7 \\ 20 \\ -75 \end{bmatrix} = \begin{bmatrix} -75 \\ 19 \\ 7 \\ 20 \end{bmatrix}$$

변수 벡터 $\mathtt{y}$의 값은 다음과 같다.

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix} = \begin{bmatrix} -75 \\ 44 \\ 35 \\ 80/11 \end{bmatrix}$$

상하행렬 분해를 이용하여 구한 변수 벡터 $\mathtt{x}$는 다음과 같다.

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} 2 \\ 10 \\ 3 \\ -5 \end{bmatrix}$$

4.18

```
Command Window
>> A = [1 5 -1 6;2 -1 1 -2;-1 4 -1 3;3 -7 -2 1];
>> [L,U,P] = lu(A)
L =
    1.0000         0         0         0
    0.3333    1.0000         0         0
    0.6667    0.5000    1.0000         0
   -0.3333    0.2273   -0.6364    1.0000
U =
    3.0000   -7.0000   -2.0000    1.0000
         0    7.3333   -0.3333    5.6667
         0         0    2.5000   -5.5000
         0         0         0   -1.4545
P =
     0     0     0     1
     1     0     0     0
     0     1     0     0
     0     0     1     0
```

4.19

```
Command Window
>> A = [1 5 -1 6;2 -1 1 -2;-1 4 -1 3;3 -7 -2 1];
>> c = [19;7;20;-75];
>> [L,U,P] = lu(A);
>> d = P*c;
>> y = L\d;
>> x = U\y
x =
    2.0000
   10.0000
    3.0000
   -5.0000
```

4.20

선형시스템 $\mathbf{Ax}=\mathbf{c}$의 변수 $\mathbf{x}$를 풀기 위해서 역행렬 $\mathbf{x}=\mathbf{A}^{-1}\mathbf{c}$을 이용하면 컴퓨터 연산 횟수(operation count)는 n^2이 필요하다. 그러나 상하행렬 분해에서 $\mathbf{Ly}=\mathbf{c}$를 풀고 다시 $\mathbf{Ux}=\mathbf{y}$를 푸는 경우에는 연산 횟수는 $3n$이 필요하다.

이런 이유로 상하행렬 분해를 이용하는 것이 보다 빠른 연산의 결과를 얻을 수 있다.

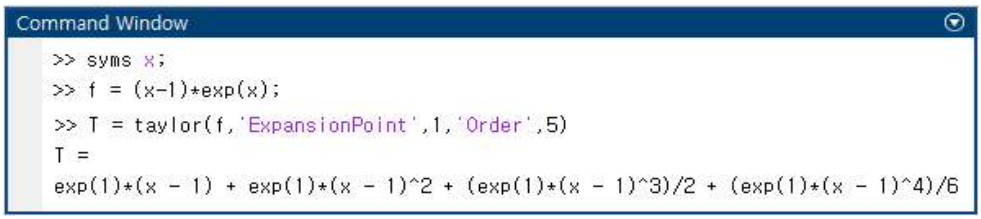
Chapter 05 연습문제

5.1

$$(x-1)e^x = 0+(x-1)e+\frac{(x-1)^2}{2!}2e+\frac{(x-1)^3}{3!}3e+\frac{(x-1)^4}{4!}4e$$
$$=(x-1)e+(x-1)^2e+\frac{(x-1)^3}{2}e+\frac{(x-1)^4}{6}e$$

5.2.

테일러 급수의 첫 번째 항이 0이므로 처음 항 네 개까지 포함시키는 경우는 5까지 지정한다.

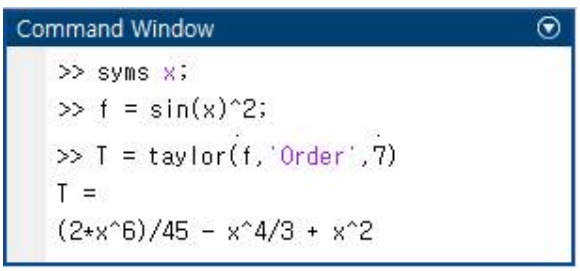


5.3

$$\sin^2 x = 0+0+\frac{x^2}{2!}\times 2+0-\frac{x^4}{4!}\times 8+0+\frac{x^6}{6!}\times 32 = x^2-\frac{x^4}{3}+\frac{2x^6}{45}$$

5.4

테일러 급수의 몇 개 항이 0이므로 처음 항 세 개까지 포함시키는 경우는 7까지 지정한다.



5.5

다음 그림에서 보여주듯이 MATLAB을 이용해서 구한 2차 테일러 다항식 근삿값은 다음과 같다.

$p_2(x) = x + x^2$

```
Command Window
>> syms x;
>> f = exp(x)*sin(x);
>> p2 = taylor(f,'Order',3)
p2 =
x^2 + x
```

식 (5.12)를 이용하여 구한 오차는 다음과 같다.

$$e^x \sin(x) - p_2(x) = \frac{x^3}{3!} f^{(3)}(\alpha)$$

주어진 경계 사이에서 최댓값은 $\alpha = 0$에서 얻게 되어 절대 오차는 다음과 같이 된다.

$$\left| e^x \sin(x) - p_2(x) \right| \le \frac{\pi^3}{384} \left| 2e^0(\cos(0) - \sin(0)) \right| = 0.1615$$

5.6

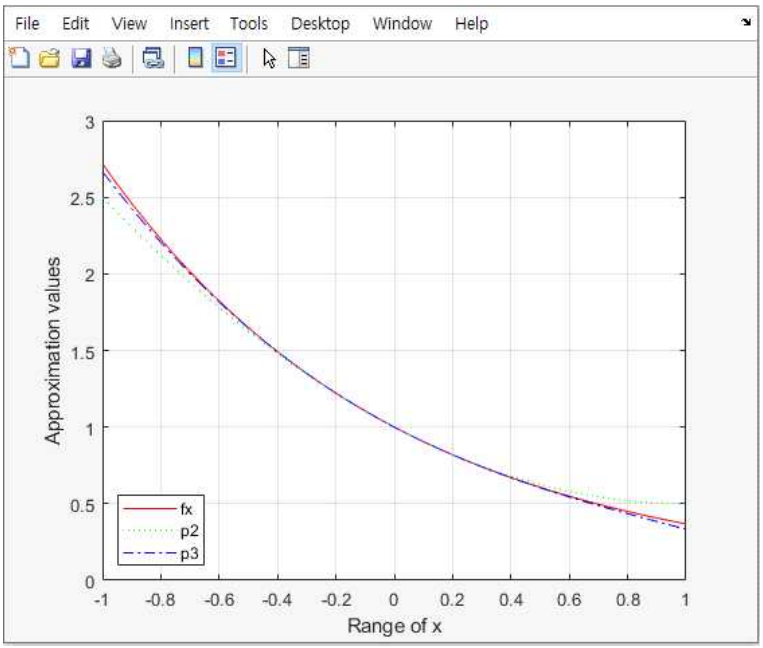
$$p_1(x) = f(0) + xf'(0) = 1 - x$$

$$p_2(x) = p_1(x) + \frac{x^2}{2!} f''(0) = 1 - x + \frac{x^2}{2}$$

$$p_3(x) = p_2(x) + \frac{x^3}{3!} f^3(0) = 1 - x + \frac{x^2}{2} - \frac{x^3}{6}$$

```
Command Window
>> x = -1.0:0.05:1.0;
>> fx = exp(-x);
>> p1 = 1-x;
>> p2 = p1+x.^2/2;
>> p3 = p2-x.^3/6;
>> plot(x,fx,'-r',x,p2,':g',x,p3,'-.b');
>> legend('fx','p2','p3','location','sw');
>> xlabel('Range of x');
>> ylabel('Approximation values');
>> grid on;
```

그림에서 **legend**로 표시한 **fx**는 근사시키는 함수 $f(x)=\exp(-x)$, **p2**는 2차 테일러 다항식 근삿값 $p_2(x)$ 그리고 **p3**은 3차 테일러 다항식 근삿값 $p_3(x)$를 각각 나타내고 있다.



다음 그림은 MATLAB을 실행하여 2차 및 3차 테일러 다항식에 대한 절대 오차의 최댓값을 보여준다.

```
Command Window
>> x = -1.0:0.05:1.0;
>> fx = exp(-x);
>> p1 = 1-x;
>> p2 = p1+x.^2/2;
>> p3 = p2-x.^3/6;
>> format long;
>> M2 = max(abs(p2-fx))
M2 =
   0.218281828459046
>> M3 = max(abs(p3-fx))
M3 =
   0.051615161792379
```

5.7

3차 테일러 다항식 근삿값은 다음과 같다.

$$p_3(x) = f(a) + (x-a)f'(a) + \frac{(x-a)^2}{2!}f''(a) + \frac{(x-a)^3}{3!}f^3(a)$$

따라서 최종적으로 구하는 결과를 부동 소수점 다섯 자리까지 나타내면 다음과 같다.

$$\sqrt[3]{9} \approx p_3(9) = 2 + \frac{1}{12}(1) - \frac{1}{288}(1)^2 + \frac{5}{20736}(1)^3 = 2.08010$$

5.8

식 (5.20)을 이용하여 구하는 경계 오차는 다음과 같다.

$$\text{경계 오차} = \left|\sqrt[3]{9} - p_3(9)\right| \le \max\frac{(9-8)^4}{4!}\left|f^4(\alpha)\right| = \frac{27}{8} \times \left|\frac{80}{81} \times (9)^{-11/3}\right|$$

$$\text{경계 오차} \le 0.00106$$

실제 오차는 계산기를 사용하여 부동 소수점 다섯 자리까지 계산된 $\sqrt[3]{9}$의 값 2.08008에서 [연습문제 5.7]에서 근삿값 2.08010을 빼준 다음과 같은 결과를 얻는다.

$$\text{실제 오차} = \max\left|f(9) - p_3(9)\right| = \left|2.08008 - 2.08010\right| = 0.00002$$

5.9

$a = -1$인 점에서 3차 테일러 다항식은 다음과 같다.

$$p_3(x) = f(-1) + (x+1)f'(-1) + \frac{(x+1)^2}{2!}f''(-1) + \frac{(x+1)^3}{3!}f^3(-1)$$

$$p_3(x) = 15 - 36(x+1) + 47(x+1)^2 - 34(x+1)^3 = -34x^3 - 55x^2 - 44x - 8$$

그리고 경계 오차는 다음과 같다.

$$\left|\text{Error}\right| \le \max_{x\in[-2,2]}\frac{(x+1)^4}{4!}\left|f^4(\alpha)\right| = \frac{1}{24}f^4(-2)$$

여기서 4차 미분항은 $f^4(x) = -240x + 72$이다.

5.10

함수 $f(x)$는 7차항 다항식이기 때문에 8번째 미분 항부터 나머지 미분 항까지는 0이 되어 결국 25차항 테일러 다항식은 7차항 테일러 다항식과 같게 된다. 즉 $p_{25}(x) = p_7(x)$이다. 그림에서 호너의 방법을 이용하여 각 항에 대한 계수들을 구하는 것을 보여주고 있다.

호너의 방법을 이용한 최종 결과는 다음과 같다.

$$p_{25}(x) = 20 + 68(x-1) + 134(x-1)^2 + 164(x-1)^3 + 128(x-1)^4 + 62(x-1)^5 + 17(x-1)^6 + 2(x-1)^7$$

Chapter 06 연습문제

6.1

(a) 주어진 자료점을 함수에 대입하고 각각의 계수를 구하면 다음과 같다.

$$\alpha_1 = 2, \quad \alpha_2 = -1, \quad \alpha_3 = -4$$

그러므로 구하는 함수는 $P(x) = 2 - \cos(\pi x) - 4\sin(\pi x)$가 된다.

(b) 본문의 식 (6.6)과 식 (6.7)을 이용하여 푸는 문제이다.

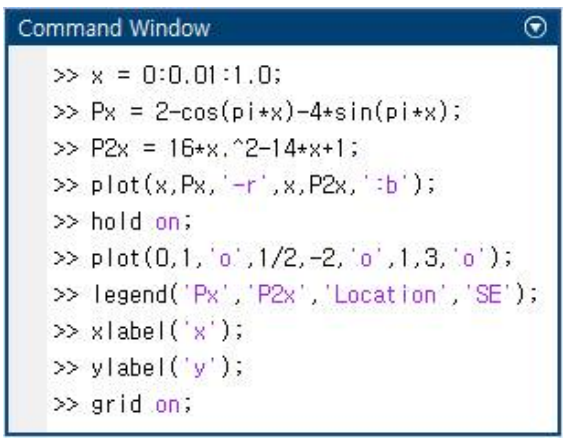
주어진 자료점 $(x_0 = 0, y_0 = 1)$, $\left(x_1 = \frac{1}{2}, y_1 = -2\right)$, $(x_2 = 1, y_2 = 3)$을 대입하면

$$P_2(x) = (1)\frac{\left(x-\frac{1}{2}\right)(x-1)}{\left(0-\frac{1}{2}\right)(0-1)} + (-2)\frac{(x-0)(x-1)}{\left(\frac{1}{2}-0\right)\left(\frac{1}{2}-1\right)} + (3)\frac{(x-0)\left(x-\frac{1}{2}\right)}{(1-0)\left(1-\frac{1}{2}\right)}$$

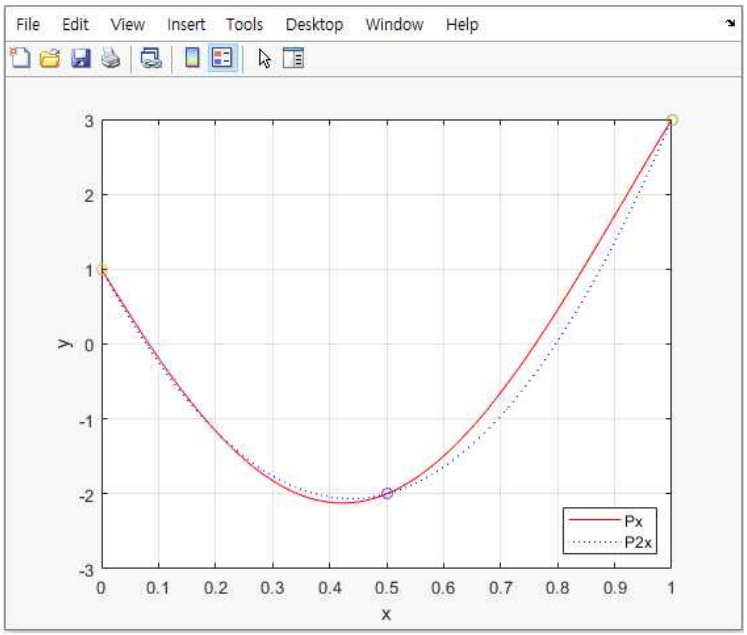
$$P_2(x) = 16x^2 - 14x + 1$$

6.2

다음 그림은 MATLAB 그래프를 그리기 위한 입력 화면을 나타낸다.



다음 그림은 실행 결과의 그래프를 나타낸다.



6.3

본문의 식 (6.3b)에 y_0과 y_1 대신에 $f(x_0)$과 $f(x_1)$을 대입하여 풀면 다음의 식을 얻는다.

$$P_1(x) = \frac{(x_1 - x)f(x_0) + (x - x_0)f(x_1)}{x_1 - x_0}$$

문제 풀이를 쉽게 하기 위해서 위의 식 분자 부분에 $(x - x_0)f(x_0)$을 한 번씩 더하고 빼준다.

$$P_1(x) = \frac{(x_1 - x)f(x_0) + (x - x_0)f(x_0) - (x - x_0)f(x_0) + (x - x_0)f(x_1)}{x_1 - x_0}$$

분자의 항들을 다시 정리하면 다음과 같은 과정을 보인다.

$$P_1(x) = \frac{(x_1 - x_0)f(x_0) + (x - x_0)[f(x_1) - f(x_0)]}{x_1 - x_0}$$

$$P_1(x) = f(x_0) + \frac{x - x_0}{x_1 - x_0}\left[f(x_1) - f(x_0)\right]$$

정의된 $\xi = \frac{x - x_0}{x_1 - x_0}$를 대입하면 다음과 같은 결과를 얻는다.

$$P_1(x) = f(x_0) + \xi\left[f(x_1) - f(x_0)\right]$$

6.4

식 (6.3d)에 의해 자료점과 이에 상응하는 y값 $(x_0 = 3, y_0 = 1.0986)$, $(x_1 = 4, y_1 = 1.3863)$과 추정하려는 x 값 3.5를 직접 대입하면 다음의 결과를 얻는다.

$$\ln 3.5 = 1.0986 + \left(\frac{1.3863 - 1.0986}{4 - 3}\right)(3.5 - 3) = 1.1925$$

경계 오차는 본문의 식 (6.39)를 이용해서 구하면 다음과 같다.

$$f(x) - P_1(x) = (x - x_0)(x - x_1)\frac{f''(c_x)}{2!} \qquad (1)$$

식 (1)에 자료점을 직접 대입하여 풀면 다음과 같다.

$$f(3.5) - P_1(3.5) = (3.5 - 3)(3.5 - 4)\frac{f''(c_x)}{2!}$$

범위 $3 \le c_x \le 4$에서 함수 $f(x) = \ln x$에 대한 2차 미분값은 $c_x = 3$인 값에서 최대이다. 그러므로 최종 구하는 경계 오차는 다음과 같이 계산된다.

$$\left|\ln 3.5 - P_1(3.5)\right| \le (3.5 - 3)(3.5 - 4)\left|\frac{1}{(3)^2(2)}\right| = 0.0139$$

6.5

2 차 보간법을 이용하는 경우에 세 개의 자료점이 필요하다. 추정하려는 x의 값 3.5를 추정하기 위해서 다음 자료점 세 개를 이용한다.

$$(x_0=2, y_0=0.6931), \quad (x_1=3, y_1=1.0986), \quad (x_2=4, y_2=1.3863)$$

식 (6.7)을 이용하여 주어진 자료점과 $x=3.5$를 대입하면 다음과 같다.

$$L_0(3.5)=\frac{(3.5-3)(3.5-4)}{(2-3)(2-4)}=-0.125$$
$$L_1(3.5)=\frac{(3.5-2)(3.5-4)}{(3-2)(3-4)}=0.75$$
$$L_2(3.5)=\frac{(3.5-2)(3.5-3)}{(4-2)(4-3)}=0.375$$

식(6.6)에 대입하면 다음과 같은 결과를 나타낸다.

$$P_2(3.5)=0.6931\times(-0.125)+1.0986\times 0.75+1.3863\times 0.375=1.2572$$

경계 오차는 본문의 식 (6.42)를 이용하여 구한다. 식 (6.42)을 다시 쓰면 다음과 같다.

$$f(x)-P_2(x)=(x-x_0)(x-x_1)(x-x_2)\frac{f^{(3)}(c_x)}{3!} \qquad (1)$$

주어진 함수 $f(x)=\ln x$에 대한 1차부터 3차까지의 미분은 다음과 같다.

$$f'(x)=\frac{1}{x},\ f''(x)=-\frac{1}{x^2},\ f^{(3)}(x)=\frac{2}{x^3}$$

식 (1)에 자료점을 직접 대입하여 풀면 다음과 같다.

$$f(3.5)-P_2(3.5)=(3.5-2)(3.5-3)(3.5-4)\frac{f^{(3)}(c_x)}{3!}$$

범위 $2\le c_x\le 4$에서 함수 $f(x)=\ln x$에 대한 3차 미분값은 $c_x=2$의 값에서 최대가 된다. 그러므로 최종 구하는 경계 오차는 다음과 같이 계산된다.

$$\left|\ln 3.5-P_2(3.5)\right|\le(3.5-2)(3.5-3)(3.5-4)\left|\frac{1}{(2)^3(6)}\right|=0.0156$$

6.6

라그랑주 공식을 이용하여 2차 보간 다항식을 구하면 다음과 같다.

$$P_2(x) = f(2)L_0(x) + f(3)L_1(x) + f(4)L_2(x)$$

$$P_2(x) = 0.6931 \times \frac{(x-3)(x-4)}{(2-3)(2-4)} + 1.0986 \times \frac{(x-2)(x-4)}{(3-2)(3-4)} + 1.3863 \times \frac{(x-2)(x-3)}{(4-2)(4-3)}$$

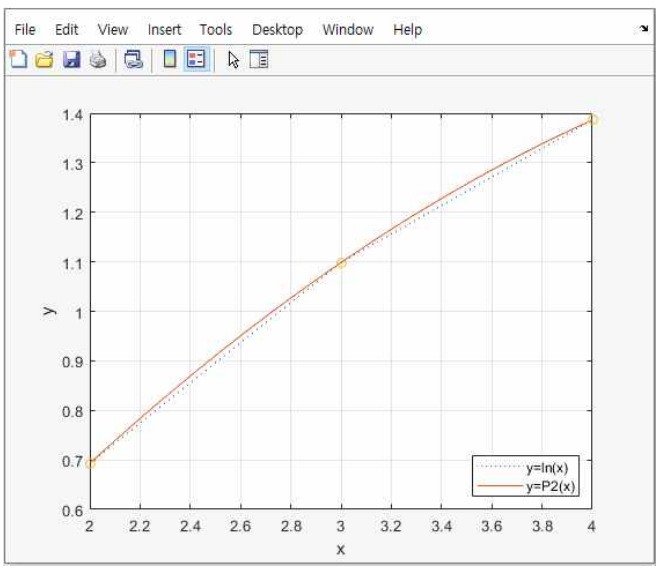
$$P_2(x) = -0.0589x^2 + 0.7x - 0.4713$$

MATLAB 스크립트 파일은 다음과 같다.

FigP6_6.m

```
% Problem 6.6
x = [2,3,4];
y = [0.6931,1.0986,1.3863];
a1 = [-0.0589,0.7,-0.4713];
x1 = 2:0.01:4;
y1 = log(x);
f1 = polyval(a1,x1);
plot(x,y1,':');
hold on;
plot(x1,f1);
plot(x,y,'o');
xlabel('x');
ylabel('y');
legend('y=ln(x)','y=P2(x)','Location','SE');
grid on;
```

MATLAB 스크립트 파일을 실행하여 그려진 그래프는 다음과 같다.



6.7

자료점 $(x_0, f_0), (x_1, f_1), \cdots, (x_n, f_n)$에 대한 라그랑주 다항식은

$$P_n(x) = f(x_0)L_0(x) + f(x_1)L_1(x) + \cdots + f(x_n)L_n(x)$$

또는 다음과 같이 나타낼 수 있다.

$$P_n(x) = \sum_{i=0}^{n} f(x_i)L_i(x) \qquad (1)$$

$f(x) = 1$이면 식 (1)은 다음과 같이 다시 쓸 수 있다.

$$P_n(x) = \sum_{i=0}^{n} L_i(x) \qquad (2)$$

경계 오차를 이용하면 다음을 얻는다.

$$f(x) = P_n(x) + (x - x_0)\cdots(x - x_n)\frac{f^{(n+1)}(\xi)}{(n+1)!} \qquad (3)$$

상수 함수 $f(x) = 1$의 모든 미분항은 $f'(\xi) = f''(\xi) = \cdots = f^{(n+1)}(\xi) = 0$이므로 식 (3)은 다음과 같이 쓸 수 있다.

$$1 - P_n(x) = 0 \text{ 또는 } P_n(x) = 1$$

그러므로 최종 결과는 다음과 같다.

$$\sum_{i=0}^{n} L_i(x) = 1$$

6.8

주어진 자료로부터, 네 개 자료점은 다음과 같이 나타낼 수 있다.

$x_0 = 0,\ x_1 = 1,\ x_2 = 2,\ x_3 = 3$

네 개의 자료점을 포함하는 라그랑주 공식은 다음과 같이 쓸 수 있다.

$$P_3(x) = f(0)L_0(x) + f(1)L_1(x) + f(2)L_2(x) + f(3)L_3(x)$$

식에 $x = \frac{1}{2}$을 대입하면 $f\left(\frac{1}{2}\right)$의 근사치는 다음과 같다.

$$f\left(\frac{1}{2}\right) \approx P_3\left(\frac{1}{2}\right) = \frac{1}{6}\left(\frac{1}{2}-2\right)\left(\frac{1}{2}-3\right)\left(\frac{1}{2}+2\right) = \frac{25}{16}$$

6.9

$$\left|f(3) - P_3(3)\right| \le \frac{\left|(3-0)(3-1)(3-2)(3-4)\right|}{4!} M_4 = \frac{M_4}{4}$$

6.10

$$m = \frac{2}{h} + 1 + 708.2 \approx 708$$

6.11

$$m = \frac{2}{h} + 1 + 101.8 \approx 102$$

6.12

(a) 구간적 선형 보간 함수는 다음과 같다.

$$l(x)=\begin{cases} x+1 & (1\le x\le 2) \\ 3 & (2\le x\le 3) \end{cases}$$

(b) 구간적 2 차 보간 함수는 다음과 같다.

$$q(x)=-0.5x^2+2.5x, \qquad (1\le x\le 3)$$

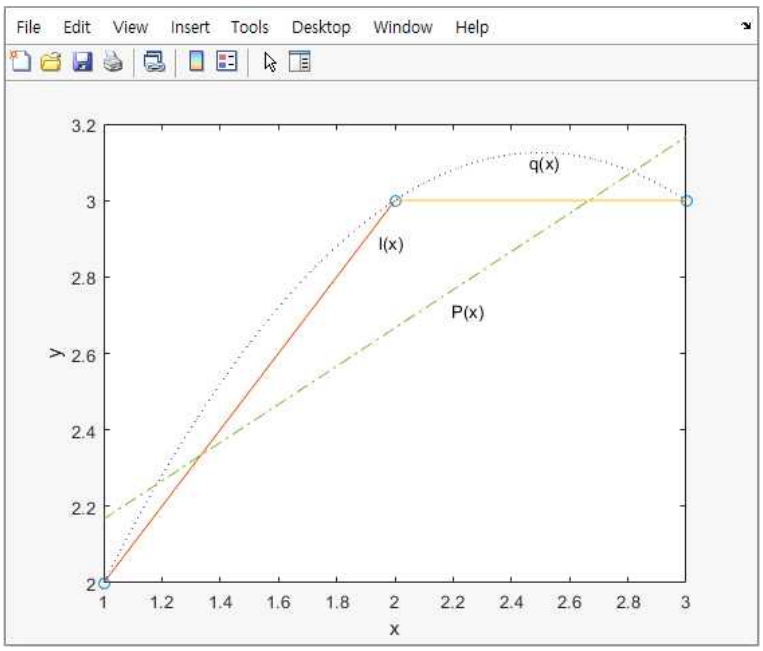
6.13

MATLAB 스크립트 파일은 다음과 같다.

FigP6_13.m

```
x = [1,2,3];
y = [2,3,3];
x1 = 1:0.01:2;
x2 = 2:0.5:3;
lx1 = x1+1;
lx2 = [3 3 3];
qx=[-0.5,2.5,0];
x3 = 1:0.01:3;
f = polyval(qx,x3);
Px = (1/2)*x3+(5/3);
plot(x,y,'o')
hold
plot(x1,lx1,x2,lx2);
plot(x3,f,':',x3,Px,'-.');
xlabel('x');
ylabel('y');
gtext('l(x)');
gtext('q(x)');
gtext('P(x)');
```

MATLAB 스크립트 파일을 실행하여 그려진 그래프는 다음과 같다.

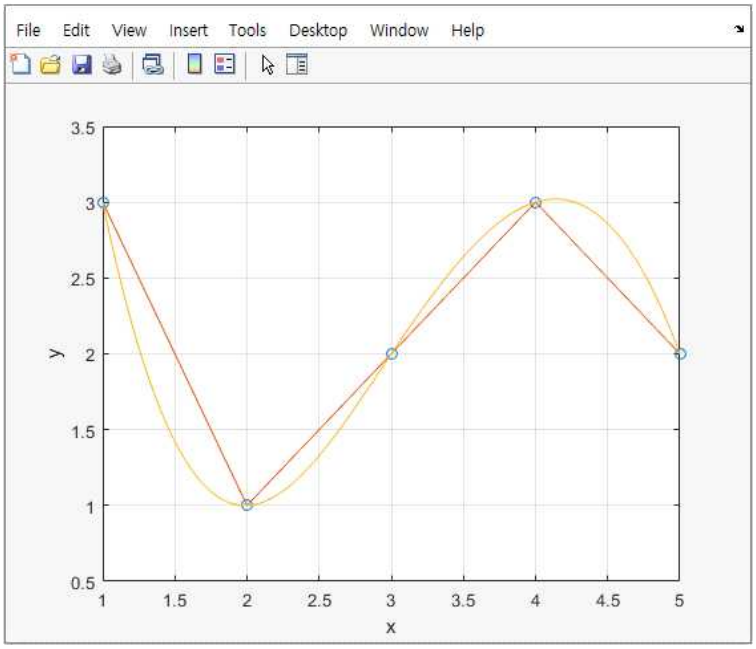


6.14

입력 화면은 다음과 같다. interp1 명령어는 선형 보간 함수의 그래프를 spline 명령어는 스플라인 함수의 그래프를 그릴 수 있다.

```
Command Window
>> x = [1,2,3,4,5];
>> y = [3,1,2,3,2];
>> x_int = 1:0.01:5;
>> y1 = interp1(x,y,x_int);
>> y_int = spline(x,y,x_int);
>> plot(x,y,'o',x_int,y1,x_int,y_int);
>> xlabel('x');
>> ylabel('y');
>> grid on;
```

입력 화면을 실행한 결과는 다음과 같다.



6.15

구하는 세 개 변수는 $\alpha_1 = \frac{1}{2}$, $\alpha_2 = \frac{3}{2}$, $\alpha_3 = -\frac{1}{2}$이고 최종 근사시킨 함수는 다음과 같다.

$$P(x) = \frac{1}{2} + \frac{3}{2}\sin(x) - \frac{1}{2}\cos(x)$$

Chapter 07 연습문제

7.1

먼저 초기 시작하는 두 개의 값은 $a=1.0000$과 $b=2.0000$로 놓는다.

$n=3$인 경우

$$c=\frac{a+b}{2}=\frac{1.7500+2.0000}{2}=1.8750$$

$$f(b=2.0000)=-0.6109,\quad f(c=1.8750)=-0.0710\quad\rightarrow f(b)\cdot f(c)>0$$

$$b=c=1.8750\rightarrow\widetilde{\alpha_3}$$

$$a\rightarrow a=1.7500$$

$$\left|\alpha-\widetilde{\alpha_3}\right|=0.0002$$

7.2

만일 구간 $[1,2]$에서 함수의 부호가 바뀌고 $f'(x)>0$이면 주어진 구간 내에 고유한 근이 존재하게 된다. $f(1)=1-4+1=-2$와 $f(2)=8-8+1=1$에 대해서 $f(1)\cdot f(2)<0$의 조건이 되어 함수의 부호가 바뀌고 또한 함수 $f(x)$를 미분한 $f'(x)=3x^2-4$가 주어진 구간 $[1,2]$에서 x의 증가함수가 되므로 근이 존재한다.

$n=1$인 경우

$$c=\frac{a+b}{2}=\frac{1.0000+2.0000}{2}=1.5000$$

$$f(b=2.0000)=1.0000,\quad f(c=1.5000)=-1.6250\quad\rightarrow f(b)\cdot f(c)<0$$

$$a=c=1.5000\rightarrow\widetilde{\alpha_1}$$

$$b\rightarrow b=2.0000$$

$n=2$인 경우

$$c=\frac{a+b}{2}=\frac{1.5000+2.0000}{2}=1.7500$$

$$f(b=2.0000)=1.0000,\quad f(c=1.7500)=-0.6406\quad\rightarrow f(b)\cdot f(c)<0$$

$$a=c=1.7500\rightarrow\widetilde{\alpha_2}$$

$$b\rightarrow b=2.0000$$

$n=5$인 경우

$$c=\frac{a+b}{2}=\frac{1.8125+1.8750}{2}=1.8438$$

$f(b=1.8750)=0.0918,\quad f(c=1.8438)=-0.1073\quad\rightarrow f(b)\cdot f(c)<0$

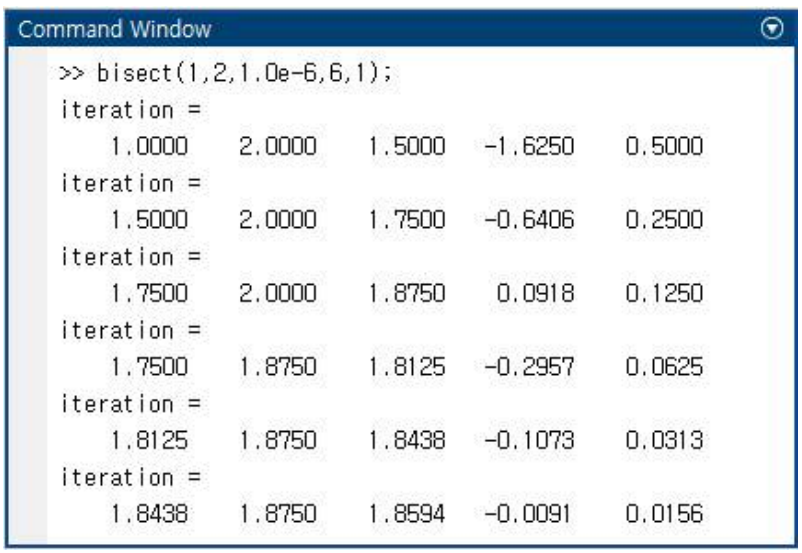
$a=c=1.8438\rightarrow\widetilde{\alpha_5}$

$b\rightarrow b=1.8750$

7.3

그림에서 MATLAB의 사용자정의 함수 bisect를 이용하여 실행된 결과를 보여주고 있다. bisect 함수의 괄호 안에 입력한 인자들은 구간의 하한값 1, 상한값 2, 허용 오차 10^{-6}, 반복 횟수 6, 그리고 사용자정의 함수에서 함수 $f(x)=x^3-4x+1$를 지정한 색인 1을 각각 나타낸다.

매번 반복하여 나온 결과들이 모두 다섯 개 열로 표현되고 있다. 이들 열은 a, b, c, $f(c)$, $b-c$를 각각 표시하고 있다. MATLAB 반복 실행은 여섯 번 하였지만, 만일 1열 한 번 반복 실행을 하면 소수점 네 자리까지 표시된 함수의 실제 근 $\alpha=1.8608$ 값을 얻는다.



7.4

$$x_1 = x_0 - \frac{x_0^3 - 4x_0 + 1}{3x_0^2 - 4} = 2.0909$$

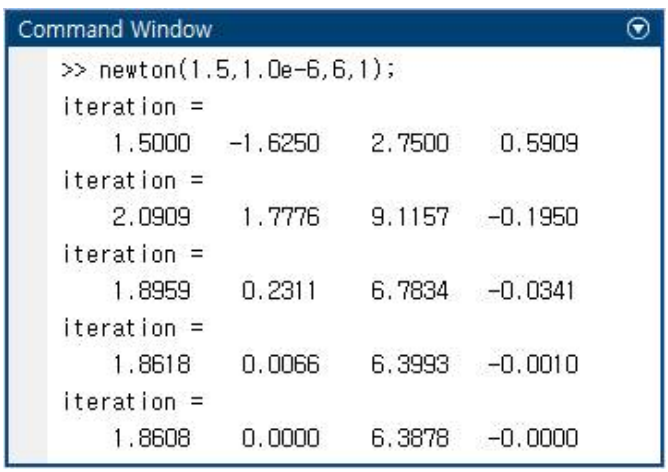
이 되고 계속해서 $n=1$, $n=2$, $n=3$을 차례로 대입하면 다음과 같은 추정하려는 각각의 근을 얻는다.

$$x_2 = x_1 - \frac{x_1^3 - 4x_1 + 1}{3x_1^2 - 4} = 1.8959$$
$$x_3 = x_2 - \frac{x_2^3 - 4x_2 + 1}{3x_2^2 - 4} = 1.8618$$
$$x_4 = x_3 - \frac{x_3^3 - 4x_3 + 1}{3x_3^2 - 4} = 1.8608$$

7.5

그림에서 MATLAB의 사용자정의 함수 **newton**을 이용하여 실행된 결과를 보여주고 있다. **newton** 함수의 괄호 안에 입력한 인자들은 초기 추정 x_0의 값, 허용 오차 10^{-6}, 반복 횟수 6, 그리고 사용자정의 함수에서 함수 $f(x)=x^3-4x+1$를 지정한 색인 번호 1을 각각 나타낸다.

매번 반복하여 나온 결과들이 모두 네 개 열로 표현되고 있다. 이들 열 들은 동일한 과정을 반복할 때마다 새롭게 계산되어 추정하는 각각의 근의 값 x_n, 함수에 x_n을 대입한 $f(x_n)$의 결과, 1차 도함수에 x_n을 대입한 $f'(x_n)$의 결과, 그리고 식 (7.9)를 이용하여 새롭게 계산된 추정값과 바로 이전의 추정값과의 차이를 표시하는 $x_n - x_{n-1}$ 를 나타낸다.



7.6

$$x_2 = x_1 - f(x_1) \cdot \frac{x_1 - x_0}{f(x_1) - f(x_0)} = 1.0000 - f(1.0000) \cdot \frac{1.0000 - 2.0000}{f(1.0000) - f(2.0000)} = 1.6667$$

반복해서 식 $n=2$와 $n=3$, $n=4$를 차례로 대입하면 다음과 같은 근들을 얻는다.

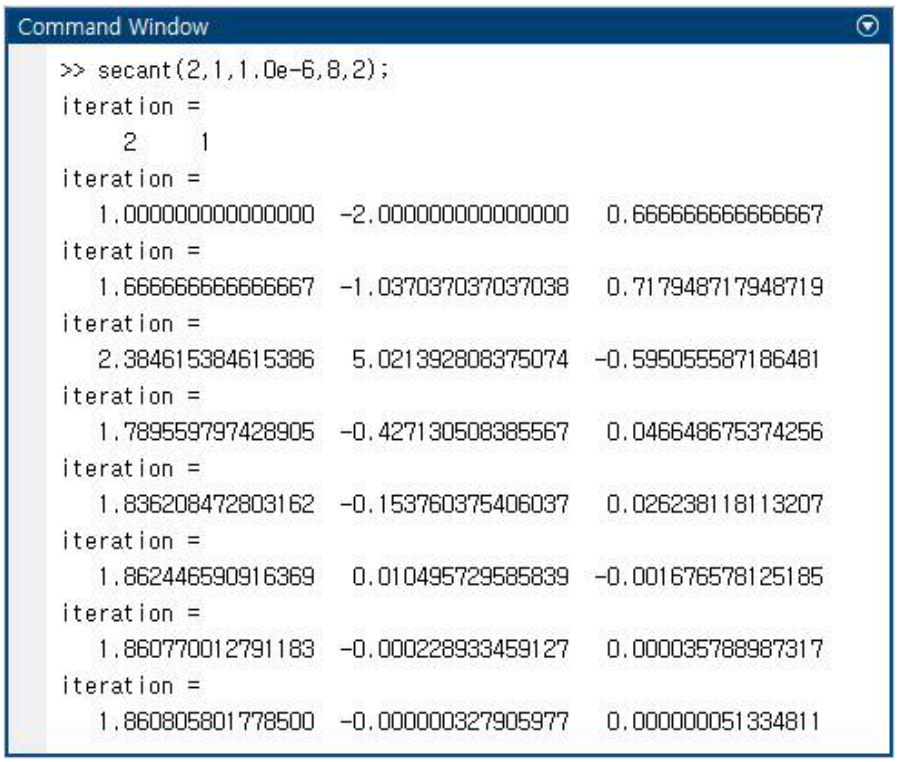
$$x_3 = x_2 - f(x_2) \cdot \frac{x_2 - x_1}{f(x_2) - f(x_1)} = 1.6667 - f(1.6667) \cdot \frac{1.6667 - 1.0000}{f(1.6667) - f(1.0000)} = 2.3846$$

$$x_4 = x_3 - f(x_3) \cdot \frac{x_3 - x_2}{f(x_3) - f(x_2)} = 2.3846 - f(2.3846) \cdot \frac{2.3846 - 1.6667}{f(2.3846) - f(1.6667)} = 1.7896$$

$$x_5 = x_4 - f(x_4) \cdot \frac{x_4 - x_3}{f(x_4) - f(x_3)} = 1.7896 - f(1.7896) \cdot \frac{1.7896 - 2.3846}{f(1.7896) - f(2.3846)} = 1.8362$$

7.7

그림에서 MATLAB의 사용자정의 함수 secant를 이용하여 실행된 결과를 보여주고 있다.



secant 함수의 괄호 안에 입력한 인자들은 초기 추정 x_0과 x_1값, 허용 오차 10^{-6}, 반복 횟수, 그리고 사용자정의 함수에서 함수 $f(x)=x^3-4x+1$를 지정한 색인 2를 각각 나타낸다.

매번 반복하여 나온 결과들이 모두 세 개 열로 표현되고 있다. 이들 열은 동일한 과정을 반복할 때마다 새롭게 계산되어 추정하는 각각의 근의 값 x_n, 함수에 x_n을 대입한 $f(x_n)$의 결과, 그리고 식 (7.17)을 이용하여 새롭게 계산된 추정값과 바로 이전의 추정값과의 차이를 표시하는 $x_{n+1}-x_n$를 나타낸다.

7.8

주어진 구간의 왼쪽 끝점 1을 a_0으로 놓고 오른쪽 끝점 2를 b_0으로 먼저 지정한다. $n=0$을 식 (7.22)와 식 (7.23)에 대입하여 계산하면 다음 결과를 얻는다.

$$f(a_0=1)=-2.0000,\ f(b_0=2)=1.0000$$

$$m_0=\frac{f(b_0)-f(a_0)}{b_0-a_0}=\frac{f(2.0000)-f(1.0000)}{2.0000-1.0000}=\frac{1.0000+2.0000}{2.0000-1.0000}=3.0000$$

$$x_0=a_0-\frac{f(a_0)}{m_0}=1.0000-\frac{-2.0000}{3.0000}=1.6667$$

$$f(a_0)\cdot f(x_0)=f(1.0000)\cdot f(1.6667)=(-2.0000)(-1.0370)>0$$

$$a_1=x_0=1.6667,\ b_1=b_0=2.0000$$

$n=1$에 대해서

$$f(a_1=1.6667)=-1.0370,\ f(b_0=2.0000)=1.0000$$

$$m_1=\frac{f(b_1)-f(a_1)}{b_1-a_1}=\frac{f(2.0000)-f(1.6667)}{2.0000-1.6667}=6.1111$$

$$x_1=a_1-\frac{f(a_1)}{m_1}=1.6667-\frac{f(1.6667)}{6.1111}=1.8364$$

$$f(a_1)\cdot f(x_1)=f(1.6667)\cdot f(1.8364)=(-1.0370)(-0.1528)>0$$

$$a_2=x_1=1.8364,\ b_2=b_1=2.0000$$

$n=2$에 대해서

$f(a_2=1.8364)=-0.1528,\ f(b_2=2.0000)=1.0000$

$$m_2=\frac{f(b_2)-f(a_2)}{b_2-a_2}=\frac{f(2.0000)-f(1.8364)}{2.0000-1.8364}=7.0450$$

$$x_2=a_2-\frac{f(a_2)}{m_2}=1.8364-\frac{f(1.8364)}{7.0450}=1.8581$$

$$f(a_2)\cdot f(x_2)=f(1.8364)\cdot f(1.8581)=(-0.1528)(-0.0175)>0$$

$a_3=x_2=1.8581,\ b_3=b_2=2.0000$

$n=3$에 대해서

$f(a_3=1.8581)=-0.0175,\ f(b_3=2.0000)=1.0000$

$$m_3=\frac{f(b_3)-f(a_3)}{b_3-a_3}=\frac{f(2.0000)-f(1.8581)}{2.0000-1.8581}=7.1685$$

$$x_3=a_3-\frac{f(a_3)}{m_3}=1.8581-\frac{f(1.8581)}{7.1685}=1.8605$$

$$f(a_3)\cdot f(x_3)=f(1.8581)\cdot f(1.8605)=(-0.0175)(-0.0020)>0$$

$a_4=x_3=1.8605,\ b_4=b_3=2.0000$

$n=4$에 대해서

$f(a_4=1.8605)=-0.0020,\ f(b_4=2.0000)=1.0000$

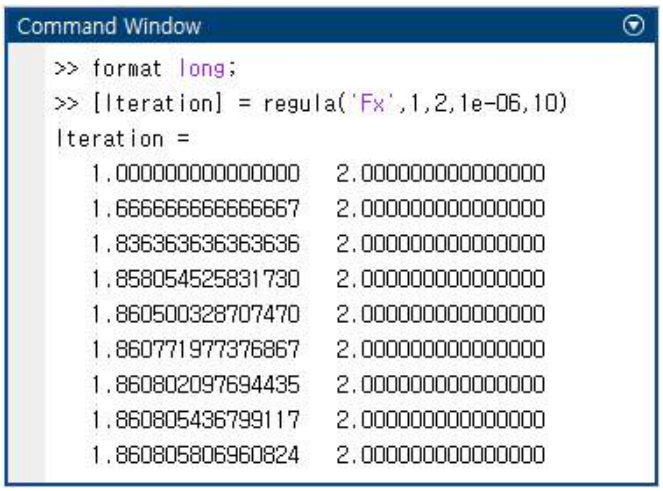
$$m_4=\frac{f(b_4)-f(a_4)}{b_4-a_4}=\frac{f(2.0000)-f(1.8605)}{2.0000-1.8605}=7.1825$$

$$x_4=a_4-\frac{f(a_4)}{m_4}=1.8605-\frac{f(1.8605)}{7.1825}=1.8608$$

소수 네 자리까지 만을 고려한 결과를 비교해 보면 다섯 번 반복 실행한 x_4의 값이 실제 근 $\alpha=1.8608$과 일치하고 있다.

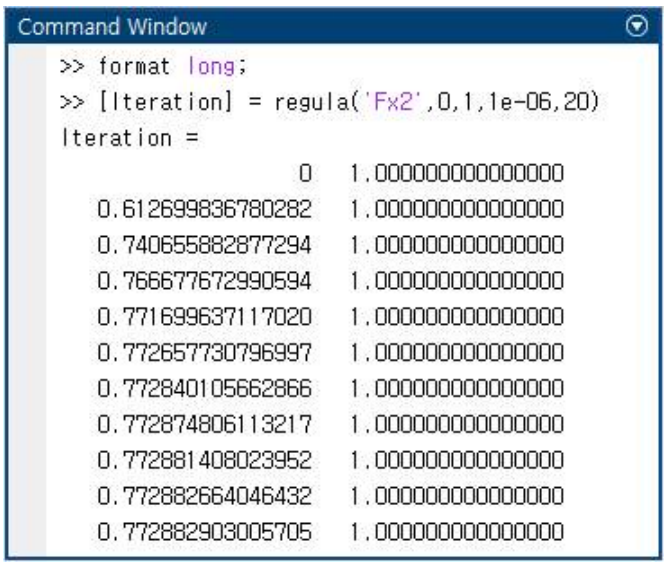
7.9

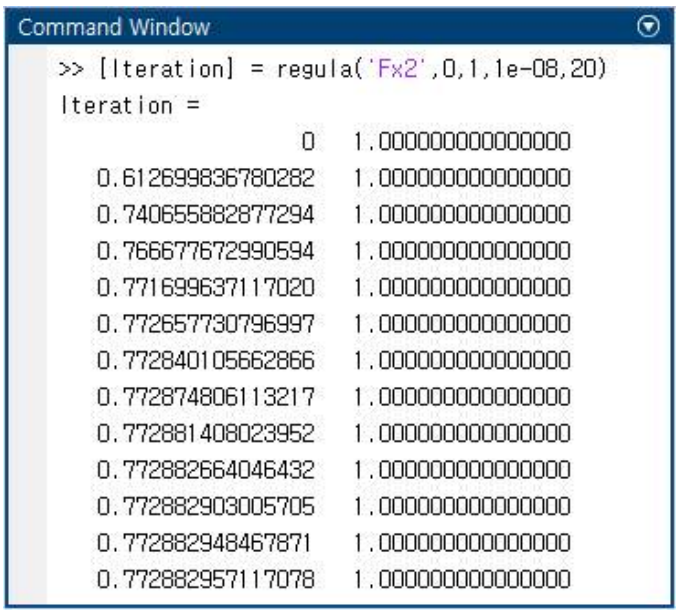
그림에서 MATLAB의 사용자정의 함수 **regula**을 이용하여 실행된 결과를 보여주고 있다. **regula** 함수의 괄호 안의 처음 인자 **Fx**는 함수 $f(x)=x^3-4x+1$에 대한 사용자정의 함수의 이름을 표시한다. **regula** 함수와는 별도로 쓰여진 사용자정의 함수 **Fx**는 **regula** 함수에서 호출이 가능하다. 1과 2는 초기 지정한 왼쪽 끝점과 오른쪽 끝점, 10^{-6}은 허용 오차, 그리고 10은 반복 횟수를 표시한다. 10회 반복 실행을 지정하였지만 여덟 번 반복의 출력 결과만을 보여 주고 있다. 여덟 번째로 계산된 결과가 실제근 1.8608005853111703과 비교하여 허용 오차 10^{-6}보다 작은 범위에 있기 때문이다.



7.10

그림에서 MATLAB의 사용자정의 함수 **regula**을 이용하여 실행된 결과를 보여준다.





regula 함수의 괄호 안의 처음 인자 **Fx2**는 함수 $f(x)=x^3-e^{-x}$에 대한 사용자정의 함수의 이름을 표시한다. **regula** 함수와는 별도로 쓰여진 사용자정의 함수 **Fx2**는 **regula** 함수에서 호출이 가능하다. 0과 1는 초기 지정한 왼쪽 끝점과 오른쪽 끝점, 10^{-6}과 10^{-8}은 실행하는 허용 오차, 그리고 20은 반복 횟수를 표시한다. 열 번 반복 실행하면 허용 오차 10^{-6}를 얻고 더 정확한 근삿값을 얻기 위하여 허용 오차를 10^{-8}로 지정하면 열 두 번 반복이 필요하다. 즉 더욱 실제 근에 가까운 정확한 근삿값을 얻기 위해서 더 많은 반복 실행이 필요하다.

7.11

실제 근은 $\alpha = 2.1544346900318 84$이고 함수 $f(x)$의 1차 도함수는 $f'(x) = 3x^2$가 된다. 초기 추정값 $x_0 = 2$과 식 (7.9)을 이용하면 다음과 같이 계산할 수 있다.

$n = 0$을 대입하면

$$x_1 = x_0 - \frac{x_0^3 - 10}{3x_0^2} = 2.166666666666667$$

$$|x_1 - x_0| = 0.166666666666667$$

$n = 1$을 대입하면

$$x_2 = x_1 - \frac{x_1^3 - 10}{3x_1^2} = 2.154503616042077$$

$$|x_2 - x_1| = 0.012163050624589$$

$n = 2$를 대입하면

$$x_3 = x_2 - \frac{x_2^3 - 10}{3x_2^2} = 2.154434692236913$$

$$|x_3 - x_2| = 0.000068923805164 < 5 \times 10^{-3}$$

7.12

식 (7.17) 을 이용하여 순서적으로 추정하려는 근들을 계산할 수 있다. 할선법의 반복적 계산 실행은 다음과 같다.

초기 두 개의 근들 사이의 절대오차는 다음과 같다.

$$|x_1 - x_0| = 1$$

$n = 1$을 대입하면

$$x_2 = x_1 - f(x_1) \cdot = \frac{x_1 - x_0}{f(x_1) - f(x_0)} = 2 - f(2) \cdot \frac{2-3}{f(2) - f(3)} = 2.105263157894737$$

$$|x_2 - x_1| = 0.105263157894737$$

$n = 2$를 대입하면

$$x_3 = x_2 - f(x_2) \cdot \frac{x_2 - x_1}{f(x_2) - f(x_1)} = 2.158194566170026$$

$$|x_3 - x_2| = 0.052931408275290$$

$n = 3$을 대입하면

$$x_4 = x_3 - f(x_3) \cdot \frac{x_3 - x_2}{f(x_3) - f(x_2)} = 2.154347659384647$$

$$|x_4 - x_3| = 0.003846906785379 < 5 \times 10^{-3}$$

7.13

사용자정의함수 nroot.m은 다음과 같다.

nroot.m

```
function xn = nroot(c, n, x0, e)
xo = x0;
j = 0;    % to count the number of iterations
xn = xo - (c-xo^n)/ (-n*(xo^(n-1))); % Newton's method
fprintf(' #t       xn #t               iterations ');
fprintf('#n ---------------------- #t #t --------------');

while (abs(xn - xo)) > e
xo = xn;
xn = xo - (c-xo^n)/ (-n*(xo^(n-1)));
j = j+1;
end

j = j+1;
fprintf('#n %20.20f     %9d#n', xn, j);
```

사용자정의함수를 실행하면 다음과 같다.

Command Window

```
>> nroot(5,2,1,1e-6);
           xn                   iterations
 ----------------------       --------------
 2.23606797749997809888            5
```

Chapter 08 연습문제

8.1

그림에서 실행 결과를 보여주고 있다. 심벌릭을 이용하면 적분의 계산 과정을 문자열로 표시하기 때문에 (25*log(5))/2−6가 최종 결과로 나왔다. 이것을 숫자로 변환하면 실제 적분값 $\int_1^5 x\ln x dx = 14.117973905426254$을 나타낸다.

```
Command Window
>> format long;
>> syms x;
>> I = int(x*log(x),1,5)
I =
(25*log(5))/2 - 6
>> fx = (25*log(5))/2 - 6
fx =
  14.117973905426254
```

8.2

$$T_4 = h\left[\frac{1}{2}f(x_0)+f(x_1)+f(x_2)+f(x_3)+\frac{1}{2}f(x_4)\right]$$
$$= h\left[\frac{\ln 1}{2}+2\ln 2+3\ln 3+4\ln 4+\frac{5\ln 5}{2}\right] = 14.250903452689032$$

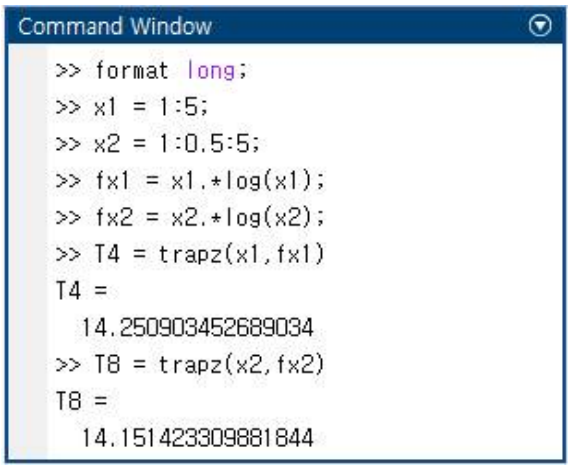
[연습문제 8.1]에서 구한 적분의 실제 값은 $\int_1^5 x\ln x dx = 14.117973905426254$이 되어 절대 오차는 0.13292955이다.

8.3

$$T_8 = h\left[\frac{1}{2}f(x_0)+f(x_1)+f(x_2)+f(x_3)+f(x_4)+f(x_5)+f(x_6)+f(x_7)+\frac{1}{2}f(x_8)\right]$$
$$= \frac{1}{2}\left[\frac{\ln 1}{2}+\frac{3}{2}\ln\frac{3}{2}+2\ln 2+\frac{5}{2}\ln\frac{5}{2}+3\ln 3+\frac{7}{2}\ln\frac{7}{2}+4\ln 4+\frac{9}{2}\ln\frac{9}{2}+\frac{5\ln 5}{2}\right]$$
$$= 14.151423309881844$$

[연습문제 8.1]에서 구한 적분의 실제 값은 $\int_1^5 x\ln x dx = 14.117973905426254$이 되어 절대 오차는 0.03354940이다.

8.4



8.5

$$n = \frac{b-a}{h} = \frac{3}{1.1547 \times 10^{-2}} \approx 260$$

8.6

$$S_4 = \frac{h}{3}\left[f(x_0)+4f(x_1)+2f(x_2)+4f(x_3)+f(x_4)\right]$$
$$= \frac{1}{3}\left[\ln 1+4\cdot 2\ln 2+2\cdot 3\ln 3+4\cdot 4\ln 4+5\ln 5\right] = 14.121583505525656$$

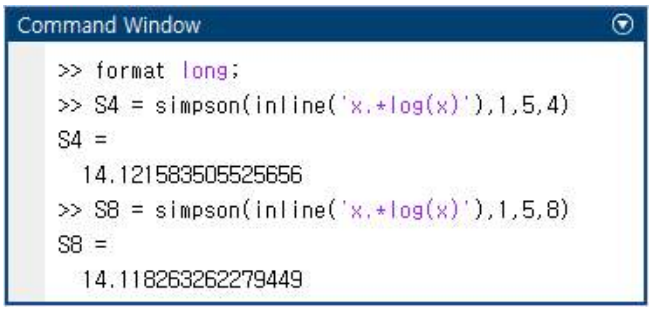
[연습문제 8.1]에서 구한 적분의 실제 값은 $\int_1^5 x\ln x dx = 14.117973905426254$이 되어 절대 오차는 0.00360960이다.

8.7

$$S_8 = \frac{h}{3}\left[f(x_0)+4f(x_1)+2f(x_2)+4f(x_3)+2f(x_4)+4f(x_5)+2f(x_6)+4f(x_7)+f(x_8)\right]$$
$$= \frac{1}{3}\left(\frac{1}{2}\right)\left[\ln 1+4\cdot\left(\frac{3}{2}\right)\ln\frac{3}{2}+2\cdot 2\ln 2+4\cdot\left(\frac{5}{2}\right)\ln\frac{5}{2}+2\cdot 3\ln 3+4\cdot\left(\frac{7}{2}\right)\ln\frac{7}{2}\right.$$
$$\left.= 2\cdot 4\ln 4+4\cdot\left(\frac{9}{2}\right)\ln\frac{9}{2}+5\ln 5\right] = 14.118263262279449$$

[연습문제 8.1]에서 구한 적분의 실제 값은 $\int_1^5 x\ln x dx = 14.117973905426254$이 되어 절대 오차는 0.00028936이다.

8.8



8.9

$$n=\frac{b-a}{h}=\frac{3}{0.2340}\approx 13$$

8.10

두 점 가우스 구적법 공식(본문 식 (8.43))을 이용하면 다음을 얻는다.

$$\int_1^5 f(x)dx\approx \frac{5-1}{2}f\left(\frac{5-1}{2}\left(-\frac{1}{\sqrt{3}}\right)+\frac{5+1}{2}\right)+\frac{5-1}{2}f\left(\frac{5-1}{2}\left(\frac{1}{\sqrt{3}}\right)+\frac{5+1}{2}\right)$$

여기서 피적분함수는 $f(x)=x\ln x$, 두 개 자료점은 $a=1$, $b=5$이다. 따라서

$$2\left(-\frac{2}{\sqrt{3}}+3\right)\ln\left(-\frac{2}{\sqrt{3}}+3\right)+2\left(\frac{2}{\sqrt{3}}+3\right)\ln\left(\frac{2}{\sqrt{3}}+3\right)=14.095598657626065$$

8.11

```
Command Window
>> format long;
>> Q1 = quad(inline('x.*log(x)'),1,5)
Q1 =
  14.117973942897050
>> Q2 = quadl(inline('x.*log(x)'),1,5)
Q2 =
  14.117973905427462
>> R = 14.117973905426254;
>> Q1_abs = abs(R-Q1)
Q1_abs =
     3.747079624361049e-08
>> Q2_abs = abs(R-Q2)
Q2_abs =
     1.207922650792170e-12
```

8.12

$$w_0 + w_1 = \int_{-1}^{1} 1dx = 2$$

$$\alpha w_0 - \alpha w_1 = \int_{-1}^{1} x dx = 0$$

$$w_0 = 1,\ \ w_1 = 1$$

8.13

(1) $f(x) = x^2$

$$\alpha^2 w_0 + \alpha^2 w_1 = \int_{-1}^{1} x^2 dx = \frac{2}{3}$$

(2) $f(x) = x^3$

$$\alpha^3 w_0 - \alpha^3 w_1 = \int_{-1}^{1} x^3 dx = 0$$

만일 $\alpha^* = \pm \frac{1}{\sqrt{3}}$의 값을 가지면 피적분 함수 $f(x) = x^2$와 $f(x) = x^3$ 에 대해서도 정확하게 된다.

8.14

간격 크기가 $h=1$인, 즉 자료점 세 개 $x_0=-1$, $x_1=0$, $x_2=1$을 이용하여 구한 심슨 공식의 결과는 다음과 같다.

$$S_2=\frac{h}{3}\left[f(x_0)+4f(x_1)+f(x_2)\right]=\frac{1}{3}\left[(-1)^4+0+(1)^4\right]=\frac{2}{3}$$

[연습문제 8.13]에서 구한 $\alpha^*=\pm\frac{1}{\sqrt{3}}$를 이용한 구적법 공식의 결과는 다음과 같다.

$$\int_{-1}^{1}f(x)dx\approx f\left(\pm\frac{1}{\sqrt{3}}\right)+f\left(\mp\frac{1}{\sqrt{3}}\right)=\left(\pm\frac{1}{\sqrt{3}}\right)^4+\left(\mp\frac{1}{\sqrt{3}}\right)^4=\frac{2}{9}$$

실제 적분의 계산은 $\int_{-1}^{1}x^4dx=\frac{2}{5}$이다. 두 개 자료점을 이용한 구적법의 절대 오차는 $\left|\frac{2}{5}-\frac{2}{9}\right|=\frac{8}{45}$, 세 개 자료점을 이용한 심슨의 절대 오차는 $\left|\frac{2}{5}-\frac{2}{3}\right|=\frac{12}{45}$가 되어 두 개 자료점을 적용한 구적법이 세 개 자료점을 적용한 심슨 공식보다 더 정확한 결과를 나타낸다.

8.15

(1) $f(x)=1$

$$w_0\cdot 1+w_1\cdot 1=\int_0^h 1\cdot dx=h$$

(2) $f(x)=x$

$$w_0\cdot 0+w_1(-h)=\int_0^h xdx=\frac{h^2}{2}$$

(1)과 (2)에서 구한 2개 선형 방정식을 풀면 w_0과 w_1를 얻을 수 있다.

$$w_0=\frac{3h}{2},\quad w_1=-\frac{h}{2}$$

오차 항은 다음과 같은 결과를 얻게 된다.

$$\int_a^{a+h}\left[f(x)-p_1(x)\right]dx=\int_a^{a+h}\underbrace{(x-a)(x-(a-h))}_{[a,a+h]\text{에서 }\geq 0}\frac{f''(\xi)}{2!}dx=\frac{5h^3}{12}f''(\xi)$$

8.16

본문 식 (8.54)를 이용하여 구간 $[1,5]$에서 각각의 부분 구간에 대한 적분의 값을 구하면 다음과 같이 된다.

$$R_{1,1}=\frac{h}{2}[f(1)+f(5)]=2[\ln 1+5\ln 5]=16.094379124341003$$

여기서 간격 크기 $h=b-a=5-1=4$가 된다.

$$R_{2,1}=\frac{h}{4}[f(1)+2f(3)+f(5)]=[\ln 1+2\cdot 3\ln 3+5\ln 5]=14.638863294179160$$

$$R_{3,1}=\frac{h}{8}[f(1)+2f(2)+2f(3)+2f(4)+f(5)]$$

$$=\frac{1}{2}[\ln 1+2\cdot 2\ln 2+2\cdot 3\ln 3+2\cdot 4\ln 4+5\ln 5]=14.250903452689032$$

$R_{1,1}$과 $R_{2,1}$을 본문 식 (8.53)에 대입하고 또한 $R_{2,1}$과 $R_{3,1}$을 본문의 식 (8.53)에 대입하면 본문 차트의 두 번째 열의 값을 구할 수 있다.

$$R_{1,2}=\frac{4^{2-1}\cdot R_{2,1}-R_{1,1}}{4^{2-1}-1}=14.153691350791879$$

$$R_{2,2}=\frac{4^{2-1}\cdot R_{3,1}-R_{2,1}}{4^{2-1}-1}=14.121583505525656$$

부분 구간 세 개에 대한 적분이기 때문에 두 번의 외삽을 이용하게 된다. 이제 $R_{1,2}$과 $R_{2,2}$을 본문 식 (8.53)에 대입하면 세 번째 열의 값을 구하게 된다.

$$R_{1,3}=\frac{4^{3-1}\cdot R_{2,2}-R_{1,2}}{4^{3-1}-1}=14.119442982507907$$

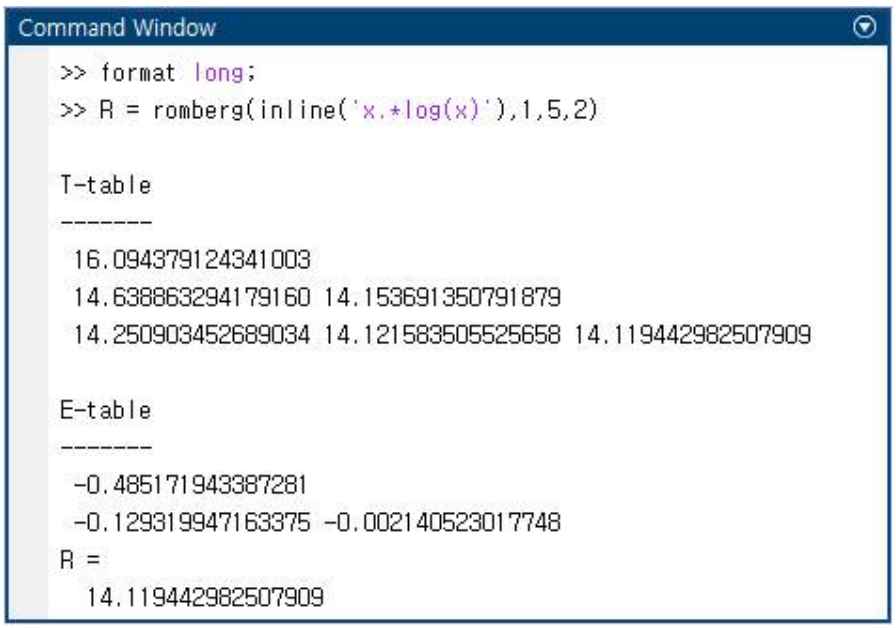
구한 값들을 본문 차트 형태로 표현하면 다음과 같이 쓸 수 있다.

16.094379124341003
14.638863294179160 14.121583505525656
14.250903452689032 14.121583505525656 14.119442982507907

실제 적분 $\int_1^5 x\ln x dx=14.117973905426254$의 결과와 비교하면 두 번의 외삽을 이용한 세 번째 열의 $R_{1,3}$값이 가장 좋은 근삿값을 보여주고 있다.

8.17

그림에서 T-table 은 [연습문제 8.16]에서 구한 롬버그 적분의 결과를 표시하고 R은 $R_{1,3}$의 값, 즉 가장 실제 적분에 근사시킨 값을 표시한다. [연습문제 8.16]에서는 풀지 않았지만, E-table 은 각각의 오차 계산 $R_{1,2}-R_{2,1}=-0.485171943387281$, $R_{2,2}-R_{3,1}=-0.129319947163376$, $R_{1,3}-R_{2,2}=-0.002140523017749$의 결과를 표시하고 있다.



Chapter 09 연습문제

9.1

Command Window

```
>> syms x h;
>> limit((sin(x+h)-sin(x))/h,h,0)
ans =
cos(x)
>> limit((cos(x+h)-cos(x))/h,h,0)
ans =
-sin(x)
```

9.2

Command Window

```
>> syms x;
>> f1 = cos(x^2);
>> D1 = diff(f1,2)
D1 =
- 2*sin(x^2) - 4*x^2*cos(x^2)
>> f2 = cos(x)^2;
>> D2 = diff(f2,2)
D2 =
2*sin(x)^2 - 2*cos(x)^2
```

9.3

```
Command Window
>> format long;
>> syms x y;
>> pdfx = diff(2*sin(x)*cos(2*y),x)
pdfx =
2*cos(2*y)*cos(x)
>> pdfy = diff(2*sin(x)*cos(2*y),y)
pdfy =
-4*sin(2*y)*sin(x)
>> x = pi/4;
>> y = pi/3;
>> pdfx = 2*cos(2*y)*cos(x)
pdfx =
  -0.707106781186547
>> pdfy = -4*sin(2*y)*sin(x)
pdfy =
  -2.449489742783178
```

9.4

$$y_1 = 1 + \left(\frac{\pi}{10}\right)(1 \cdot \cos(0)) = 1 + \frac{\pi}{10} = 1.31416$$

실제값의 계산 결과는 다음과 같다.

$$y\left(\frac{\pi}{10}\right) = e^{\sin\left(\frac{\pi}{10}\right)} = 1.36209$$

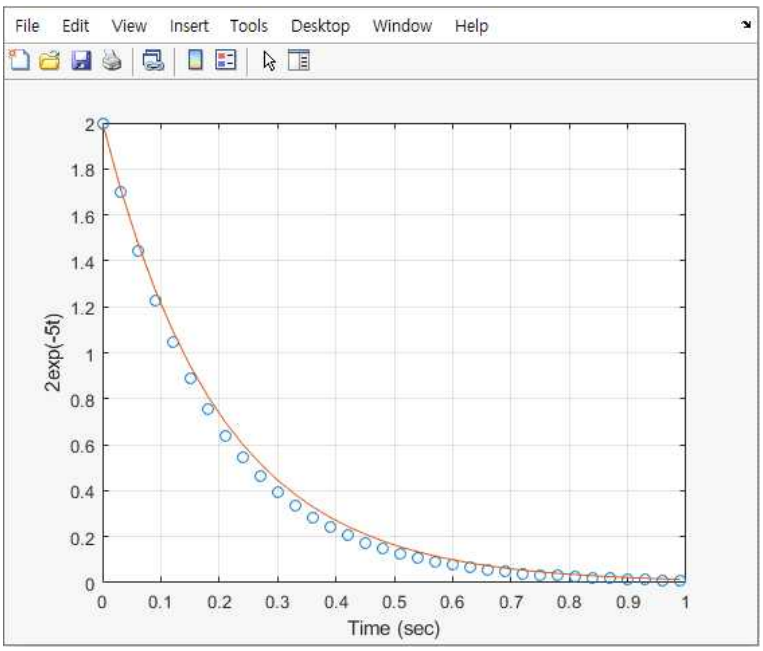
9.5

오일러 방법을 이용하여 푼 스크립트 파일을 제시하였다.

FigP9_5.m

```
h = 0.03;
y(1) = 2;
n = 0;
for t1 = h:h:1
    n = n+1;
    y(n+1) = y(n)-5*y(n)*h;
end
t = 0:h:1;
fy = 2*exp(-5*t);
plot(t,y,'o',t,fy);
xlabel('Time (sec)');
ylabel('2exp(-5t)');
axis([0 1 0 2]);
grid on;
```

초기값은 y(1) = 2 로 지정하였다. 다음 그림은 작성한 스크립트 파일의 결과를 보여주는 그래프다. 그래프에서 동그라미들의 연결은 수치적으로 풀어진 곡선을 나타내고 실선의 곡선은 실제 결과를 보여주고 있다.

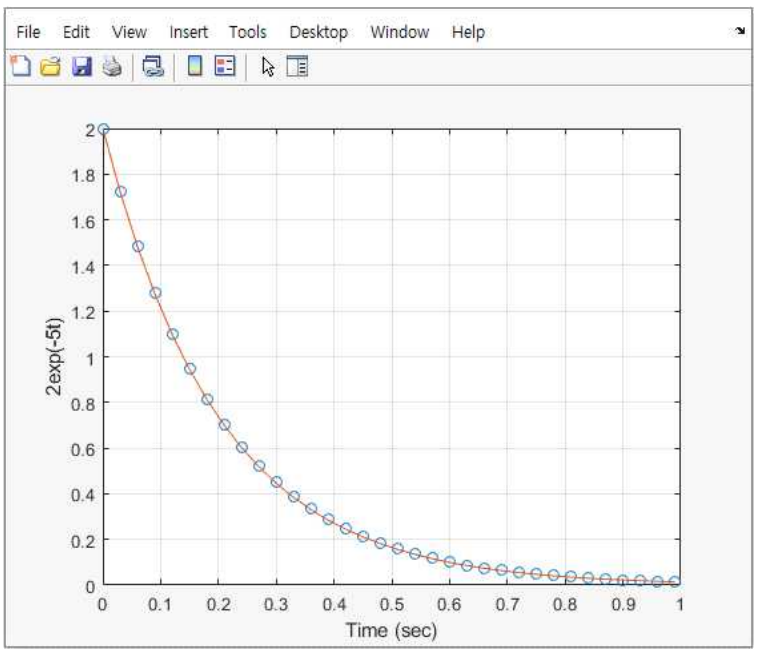


9.6

각각의 그림에서 수정된 오일러 방법을 이용하여 푼 스크립트 파일을 제시하고 수정된 오일러 방법을 실행한 그래프를 보여주고 있다. 수정된 오일러 방법으로 그려진 그래프에서 동그라미의 곡선이 원래 오일러 방법에 비해서 훨씬 실제 결과 곡선에 일치하고 있음을 알 수 있다.

FigP9_6.m

```
h = 0.03;
y(1) = 2;
n = 0;
for t1 = h:h:1
    n = n+1;
    x(n+1) = y(n)-5*y(n)*h;
    y(n+1) = y(n)+(h/2)*(-5*y(n)-5*x(n+1));
end
t = 0:h:1;
fy = 2*exp(-5*t);
plot(t,y,'o',t,fy);
xlabel('Time (sec)')
ylabel('2exp(-5t)');
axis([0 1 0 2]);
grid on;
```



9.7

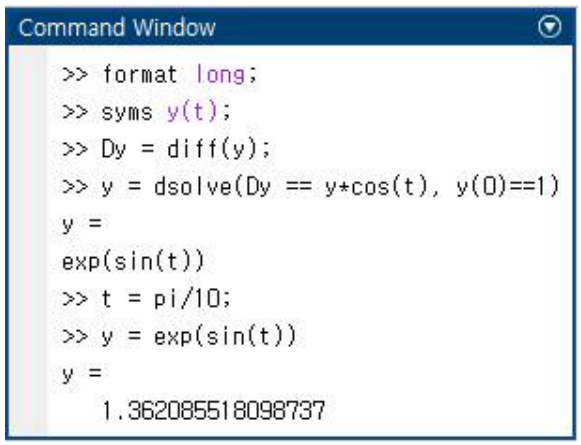
$$y_1 = 1 + \left(\frac{\pi}{10}\right)(1 \cdot \cos(0)) + \frac{\left(\frac{\pi}{10}\right)^2}{2}[(-1) \cdot \sin(0) + \cos(0) \cdot (1 \cdot \cos(0))] = 1.36351$$

실제값의 계산 결과는 다음과 같다.

$$y\left(\frac{\pi}{10}\right) = e^{\sin\left(\frac{\pi}{10}\right)} = 1.36209$$

9.8

그림에서 나타내는 결과는 미분방정식을 푼 실제값의 계산 결과이다.



9.9

$$y_1 = y_0 + \frac{h}{2}\left[K_1^{(2)} + K_2^{(2)}\right] = 1 + \left(\frac{\pi/10}{2}\right)(1 + 1.24984) = 1.35340$$

절대오차는 다음과 같다.

$$|1.36209 - 1.35340| = 8.69 \times 10^{-3}$$

9.10

$$y_1 = 1 + \left(\frac{\pi/10}{6}\right)(1 + 2\times 1.14283 + 2\times 1.16500 + 1.29913) = 1.36206$$

절대오차는 다음과 같다.

$$|1.36209 - 1.36206| = 3\times 10^{-5}$$

9.11

그래프를 그리기 위해서는 스크립트 파일을 두 개 작성한다. 그중에 하나는 다음 그림처럼 사용자정의함수로 1 차 미분방정식 ydot 을 구성하는 odecos.m 파일이다.

odecos.m

```
1   function ydot = odecos(t,y)
2 –   ydot = y*cos(t);
```

나머지 스크립트 파일은 다음 그림처럼 사용자정의함수 odecos 를 불러와서 ode23 함수와 ode45 함수를 실행하여 그래프 두 개를 그리도록 작성한다.

FigP9_11.m

```
1 –   t1 = 0:pi/10:4*pi;
2 –   subplot(2,1,1);
3 –   [t,y] = ode23('odecos', [0, 4*pi],1);
4 –   y1 = exp(sin(t1));
5 –   plot(t,y,'o',t1,y1);
6 –   xlabel('Time (sec)');
7 –   ylabel('y(t)=exp(sin(t))');
8 –   axis([0 4*pi 0 3]);
9 –   grid on;
10 –  subplot(2,1,2);
11 –  [t,y] = ode45('odecos', [0, 4*pi],1);
12 –  y2 = exp(sin(t1));
13 –  plot(t,y,'o',t1,y2);
14 –  xlabel('Time (sec)');
15 –  ylabel('y(t)=exp(sin(t))');
16 –  axis([0 4*pi 0 3]);
17 –  grid on;
18 –  gtext('ode23');
19 –  gtext('ode45');
```

그래프에서 동그라미들을 연결해보면 수치적으로 계산된 결과들을 연결하는 곡선 형태를 보여준다. 시각적으로 동그라미들이 촘촘하기 때문에 ode45 함수를 이용해서 그린 그래프가 전반적으로 ode23 함수를 이용해서 그린 그래프보다 더 매끄러운 연결을 보여준다.