

Numerical Methods HW #01

0. Show that $\Delta\left(\frac{f_a}{g_a}\right) \cong \frac{|g_a|\Delta f_a - |f_a|\Delta g_a}{|g_a|^2}$

<Solution>

HW #1

$$\Delta\left(\frac{f_a}{g_a}\right) \cong \frac{|g_a|\Delta f_a - |f_a|\Delta g_a}{|g_a|^2}$$

$$\Delta\left(\frac{f_a}{g_a}\right) = \frac{f(x)}{g(x)} - \frac{f(x_0)}{g(x_0)}$$

$$f(x) = f(x_0) + f'(x_0)\frac{(x-x_0)}{1!} + \dots$$

$$g(x) = g(x_0) + g'(x_0)\frac{(x-x_0)}{1!} + \dots$$

$$\Delta\left(\frac{f_a}{g_a}\right) = \frac{f(x_0) + f'(x_0)(x-x_0)}{g(x_0) + g'(x_0)(x-x_0)} - \frac{f(x_0)}{g(x_0)}$$

$$= \frac{g(x_0)f'(x_0) + g'(x_0)f(x_0)(x-x_0) - f(x_0)g'(x_0) - f'(x_0)g(x_0)(x-x_0)}{g(x_0)(g(x_0) + g'(x_0)(x-x_0))}$$

$$= \frac{|g(x_0)|\Delta f_a - |f(x_0)|\Delta g_a}{|g(x_0)|^2} = \frac{|g_a|\Delta f_a - |f_a|\Delta g_a}{|g_a|^2}$$

1. 아래 함수의 도함수를 주어진 점에 대해서 도함수의 정의를 이용하여 소수점 5자리까지 정확히 구하라.

(a) $f(x) = e^x \sin x + \cos^2 x$, $x = \frac{\pi}{4}$

<Solution>

도함수의 정의에 의해 $f'(x) = \lim_{h \rightarrow 0} \frac{f(x+h) - f(x)}{h}$ 이고 h 가 충분히 작으면 $\frac{f(x+h) - f(x)}{h}$ 로 근사할 수 있다.

구간 h 를 1, 1/2, 1/4, ... 로 점점 작게 해서 값을 구했다.

구간이 클 때는 error 값이 크지만 구간이 작아질수록 error 값이 0에 근접한다.

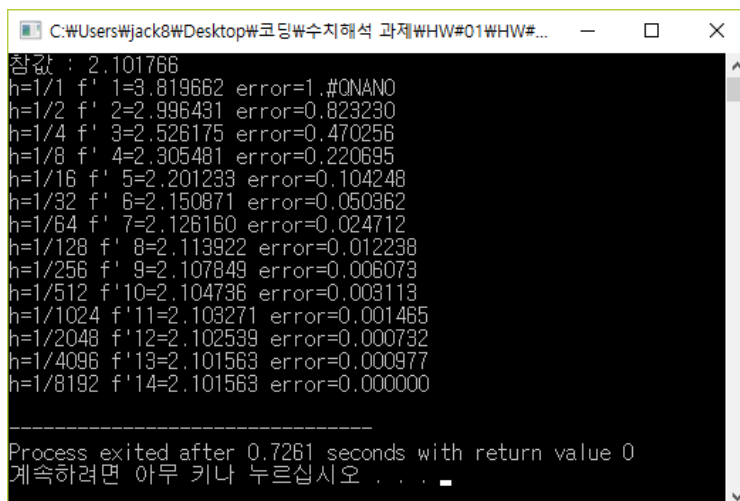
코드에서 break를 걸지 않고 계속 계산해보면 총수치오차에 의해서 오차가 진동하며 된다.

error 값은 Scarborough 판정법에 의해 0.5×10^{-5} 로 정했다.

<Code>

```
#include<stdio.h>
#include<math.h>
float f(float x)
{
    return exp(x)*sin(x)+pow(cos(x),2);
}
int main()
{
    int i;
    double error, a, b,c,d;
    double x=M_PI/4;
    error=0.5/pow(10,5);
    printf("참값 : %1f\n",exp(x)*(sin(x)+cos(x))-2*sin(x)*cos(x));
    for(i=0;i<=100000;i++)
    {
        a=f(x+1/pow(2,i))/(1/pow(2,i));
        b=f(x)/(1/pow(2,i));
        c=a-b;
        printf("h=1/%.01f f'%2d=%1f error=%1f\n",pow(2,i),i+1,c,fabs(d-c));
        if(i!=1)
        {
            if(fabs(d-c)<error)
                break;
        }
        d=c;
    }
    return 0;
}
```

<결과 창>



```
C:\Users\Wjack8\Desktop\코딩\수치해석 과제\HW#01\HW#...
참값 : 2.101766
h=1/2 f' 1=3.819662 error=1.#QNANO
h=1/4 f' 2=2.996431 error=0.823230
h=1/8 f' 3=2.526175 error=0.470256
h=1/16 f' 4=2.305481 error=0.220695
h=1/32 f' 5=2.201233 error=0.104248
h=1/64 f' 6=2.150871 error=0.050362
h=1/128 f' 7=2.126160 error=0.024712
h=1/256 f' 8=2.113922 error=0.012238
h=1/512 f' 9=2.107849 error=0.006073
h=1/1024 f' 10=2.104736 error=0.003113
h=1/2048 f' 11=2.103271 error=0.001465
h=1/4096 f' 12=2.102539 error=0.000732
h=1/8192 f' 13=2.101563 error=0.000977
h=1/16384 f' 14=2.101563 error=0.000000

-----
Process exited after 0.7261 seconds with return value 0
계속하려면 아무 키나 누르십시오 . . .
```

첫 번째 시행에서의 error값은 코드에서 d변수에 값이 아직 저장되지 않아 이렇게 표시된다.

컴퓨터를 통해 얻은 값이 참값과 소수점 셋째자리까지는 같지만 그 이후는 다르다.

참 값 =2.101766

계산 값=2.101563

(b) $f(x) = (\ln x) \sin^2 x$, $x = \frac{\pi}{4}$

<Solution>

1번 문제의 (a)와 마찬가지로 도함수정의를 구간이 매우 작을 때 $\frac{f(x+h)-f(x)}{h}$ 로 근사하여 푼다.

<Code>

```
#include<stdio.h>
#include<math.h>
float f(float x)
{
    return log(x)*pow(sin(x),2);
}
int main()
{
    int i;
    double x = M_PI/4;
    double a, error, b;
    error = 0.5/pow(10,6);
    printf("참값=%1f\n",pow(sin(x),2)/x+log(x)*2*sin(x)*cos(x));
    for(i=1;i<=100;i++)
    {
        a=(f(x+(1/pow(2,i-1)))-f(x))/(1/pow(2,i-1));
        printf("h=1/%.01f f'%2d=%1f error=%1f\n",pow(2,i-1),i,a,fabs(a-b));
        if(i!=1)
        {
            if(fabs(a-b)<error)
                break;
        }
        b=a;
    }
    return 0;
}
```

<결과 창>

```
C:\Users\Wjack8\Desktop\#코딩\수치해석 과제\HW#...
참값=0.395055
h=1/1 f' 1=0.674136 error=1.#QNAN0
h=1/2 f' 2=0.703900 error=0.029764
h=1/4 f' 3=0.586056 error=0.117844
h=1/8 f' 4=0.497867 error=0.088189
h=1/16 f' 5=0.447992 error=0.049874
h=1/32 f' 6=0.421866 error=0.026127
h=1/64 f' 7=0.408541 error=0.013325
h=1/128 f' 8=0.401817 error=0.006723
h=1/256 f' 9=0.398441 error=0.003376
h=1/512 f'10=0.396748 error=0.001694
h=1/1024 f'11=0.395897 error=0.000851
h=1/2048 f'12=0.395477 error=0.000420
h=1/4096 f'13=0.395264 error=0.000214
h=1/8192 f'14=0.395142 error=0.000122
h=1/16384 f'15=0.395020 error=0.000122
h=1/32768 f'16=0.395020 error=0.000000

-----
Process exited after 0.6872 seconds with return value 0
계속하려면 아무 키나 누르십시오 . . .
```

(a)와 마찬가지로 구간h를 줄여가며 계산을 했고 첫 번째의 error값은 코드에서 b에 저장된 값이 없기 때문에 나온 것이다.

참값과 소수점 4째자리까지는 같지만 5째자리부터는 다르다.

참 값 =0.395055

계산 값=0.395020

2. $\cos x$ 의 Maclaurin 급수는 다음과 같다.

$$\cos x = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!} + \dots$$

위의 무한급수를 이용하여 $\cos(\frac{\pi}{4})$ 의 근사값을 유효숫자 3자리까지 정확히 구하라.

<Solution>

코딩을 통해서 항을 하나씩 추가하는 방법으로 계산했다.

error값은 Scarborough 판정법에 의해 0.5×10^{-3} 으로 정했다.

<Code>

```
#include<stdio.h>
#include<math.h>
float factorial(float n)
{
    int i,x;
    x=1;
    if(i=0)
    {
        return 1;
    }
    else
    {
        for(i=1;i<=n;i++)
        {
            x=x*i;
        }
        return x;
    }
}
int main()
{
    int i;
    double x,b,c,error;
    double sum=0;
    x=M_PI/4;
    error=0.5/pow(10,3);
    printf("참값 = %lf\n",cos(x));
    for(i=0;i<=100;i++)
    {
        c=pow(-1,i)*pow(x,2*i)/factorial(2*i);
        sum=sum+c;
        printf("order=%d cos(pi/4)=%lf error=%lf\n",i,sum,c);
        if(i!=1)
        {
            if(fabs(c)<error)
            break;
        }
    }
}
```

<결과 창>

```
C:\Users\jack8\Desktop\코딩\수치해석 과제\HW#...
참값 = 0.707107
order=0 cos(pi/4)=1.000000 error=1.000000
order=1 cos(pi/4)=0.691575 error=-0.308425
order=2 cos(pi/4)=0.707429 error=0.015854
order=3 cos(pi/4)=0.707103 error=-0.000326

-----
Process exited after 0.7132 seconds with return value 0
계속하려면 아무 키나 누르십시오 . . .
```

소수점 셋째자리까지 정
확히 구하였다.

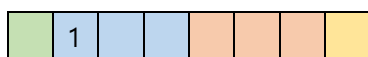
참 값 =0.707107
계산 값=0.707103

3. 버림 작업을 하는 8비트 워드 컴퓨터가 있다. 이 때 가수를 위한 비트 수는 4개이고, 지수를 위한 비트 수는 4개이다. 계산 값이 다음과 같을 때 컴퓨터에서의 저장 값을 구하라.

- (a) 1.1135 (b) 0.9501 (c) 7.7532
(d) 7.3243 (e) 0.0523 (f) 0.0685

<Solution>

8비트 워드 컴퓨터이고 가수 비트 4개 지수 비트 4개가 할당되어 있다. 가수와 지수의 부호에 대한 것도 포함한다고 하면 가수를 '표현하기 위한' 비트는 3개, 지수를 표현하기 위한 비트도 3개인 것이다.



왼쪽부터 가수의 부호, 가수, 지수, 지수의 부호를 위한 비트이다.

(a) 1.1135

$$1.1135 = 1 + 0.1135 = 1 + 0 \times \frac{1}{2} + 0 \times \frac{1}{4} + 0.1135 (\text{버림}) = (0.100)_2 \times 2^{-1} = (01000011)_2$$

01000011

(b) 0.9501

$$0.9501 = 1 \times \frac{1}{2} + 0.4501 = 1 \times \frac{1}{2} + 1 \times \frac{1}{4} + 0.2001 = 1 \times \frac{1}{2} + 1 \times \frac{1}{4} + 1 \times \frac{1}{8} + 0.0751 (\text{버림})$$

$$= (0.111)_2 \times 2^0 = (01000000)_2$$

01000000

(c) 7.7532

$$7.7532 = 1 \times 4 + 1 \times 2 + 1 \times 1 + 0.7532 (\text{버림}) = (0.111)_2 \times 2^3 = (01111110)_2$$

01111110

(d) 7.3243

$$7.3243 = 1 \times 4 + 1 \times 2 + 1 \times 1 + 0.3243 (\text{버림}) = (0.111)_2 \times 2^3 = (01111110)_2$$

01111110

(e) 0.0523

$$0.0523 \times 2^5 = 1.6736 = 1 \times 1 + 1 \times \frac{1}{2} + 0 \times \frac{1}{4} + 0.1736 (\text{버림}) = (0.110)_2 \times 2^1$$

$$0.0523 = (0.110)_2 \times 2^{1-5} = (01101001)_2$$

01101001

(f) 0.0685

$$0.0685 \times 2^5 = 2.192 = 1 \times 2 + 0.192 = 1 \times 2 + 1 \times \frac{1}{8} + 0.067 \text{ (버림)} = (0.100)_2 \times 2^2 = (01000100)_2$$

01000100

4. $\sin(x)$ 는 다음과 같이 무한급수로 정의된다.

$$\sin x = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \dots$$

이 급수를 사용하여 $\sin\left(\frac{\pi}{4}\right)$ 을 계산하는 알고리즘을 작성하라. 단, 무한급수에서 근사값에 대한 백분율 상대오차(ϵ_a)를 각각 출력하되, 그것이 0.005% 이하이면 계산을 중지하라.

<Solution>

2번 문제와 마찬가지로 항을 하나씩 추가해가면서 계산하는 프로그램을 만들었다.

<Code>

```
#include<stdio.h>
#include<math.h>
float factorial(float n)
{
    int i,x;
    x=1;
    if(i=0)
    {
        return 1;
    }
    else
    {
        for(i=1;i<=n;i++)
        {
            x=x*i;
        }
        return x;
    }
}
int main()
{
    double x,s,error;
    double sum=0;
    int i;
    x=M_PI/4;
    error=0.5/pow(10,4);
    printf("참값 = %lf\n",sin(x));
    for(i=0;i<=100;i++)
    {
        s=pow(-1,i)*pow(x,2*i+1)/factorial(2*i+1);
        sum=sum+s;
        printf("order=%d sin(pi/4)=%lf error(%)=%lf\n",i,sum,fabs(s)*100);
        if(i!=1)
        {
            if(fabs(s)<error)
            break;
        }
    }
}
```

<결과 창>

```
C:\Users\Wjack8\Desktop\코딩\수치해석 과제\HW#0...
참값 = 0.707107
order=0 sin(pi/4)=0.785398 error(%)=78.539816
order=1 sin(pi/4)=0.704653 error(%)=8.074551
order=2 sin(pi/4)=0.707143 error(%)=0.249039
order=3 sin(pi/4)=0.707106 error(%)=0.003658

-----
Process exited after 0.3597 seconds with return value 0
계속하려면 아무 키나 누르십시오 . . .
```

소수점 셋째자리까지 정확하게 구하였다.

참 값 =0.707107

계산 값=0.707106

5. 다음 무한급수

$$y = e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \dots$$

를 이용하여 e^9 와 e^{-9} 을 계산하라. 그 결과를 참값과 비교하고 만일 그 결과가 서로 일치하지 않으면 그 이유를 서술하라. 단 e^9 와 e^{-9} 의 참값은 8103.08과 1.23409×10^{-4} 이다.

<Solution>

2번이나 4번문제처럼 코드를 짤 때는 아래 결과창과 같이 14번시행부터 오차가 갑자기 커졌다. 항을 하나씩 더할 때 분자와 분모를 따로 계산해서 더하면 분자와 분모가 매우 커져 계산이 안되는 것을 보고 하나의 항을 만들 때 (x/n) 을 n 은 1부터 n 까지 곱해서 만들었다.

<Code>

```
#include<stdio.h>
#include<math.h>
int main()
{
    int i;
    double j;
    double x=9;
    double e,sum,f;
    double error=0.5/pow(10,5);
    sum=0;
    printf("참값 = %lf\n",pow(M_E,x));
    for(i=0;i<=100;i++)
    {
        f=1;
        if(i==0)
        {
            f=1;
        }
        else
        {
            for(j=1;j<=i;j++)
            {
                e=x/j;
                f=f*e;
            }
            sum=sum+f;
            printf("#%2d e^%d=%lf error=%lf\n",i+1,(int)x,sum,f);
            if(fabs(f)<error) break;
        }
    }
}
```

<결과 창>

```
C:\Users\jack8\Desktop\코딩\수치해석 과제\HW#01\HW#01_5.exe
참값 = 8103.083928
#0 e^9=1.000000 sum=1.000000 error=1.000000
#1 e^9=9.000000 sum=10.000000 error=9.000000
#2 e^9=40.500000 sum=50.500000 error=40.500000
#3 e^9=121.500000 sum=172.000000 error=121.500000
#4 e^9=273.375000 sum=445.375000 error=273.375000
#5 e^9=492.075000 sum=937.450000 error=492.075000
#6 e^9=738.112500 sum=1675.562500 error=738.112500
#7 e^9=949.001786 sum=2624.564286 error=949.001786
#8 e^9=1067.627009 sum=3692.191295 error=1067.627009
#9 e^9=1067.627009 sum=4759.818304 error=1067.627009
#10 e^9=960.864308 sum=5720.682612 error=960.864308
#11 e^9=786.161707 sum=6506.844318 error=786.161707
#12 e^9=589.621280 sum=7096.465598 error=589.621280
#13 e^9=1315.629108 sum=8412.094706 error=1315.629108
#14 e^9=17887.233186 sum=26299.327892 error=17887.233186
#15 e^9=102724.194586 sum=129023.522478 error=102724.194586
#16 e^9=924573.490190 sum=1053597.012668 error=924573.490190
#17 e^9=-57802066.487722 sum=-56748469.475054 error=57802066.487722
#18 e^9=-167062687.242671 sum=-22381156.717725 error=167062687.242671
#19 e^9=12320598574.230717 sum=12086787417.512991 error=12320598574.230717
#20 e^9=-5783490857.095395 sum=6313296560.417596 error=5783490857.095395
#21 e^9=-91555235500.643082 sum=-85241938940.225479 error=91555235500.643082
#22 e^9=-1883953293794.827400 sum=-1969195232735.053000 error=1883953293794.827400
#23 e^9=10276421218979.324000 sum=8307225986244.271500 error=10276421218979.324000
#24 e^9=-102798929829046.550000 sum=-94491703842802.281000 error=102798929829046.550000
#25 e^9=345778218515883.810000 sum=251286514673081.530000 error=345778218515883.810000
#26 e^9=-3485162813321860.500000 sum=-3233876298648779.000000 error=3485162813321860.500000
#27 e^9=39163778732752440.000000 sum=35929902434103660.000000 error=39163778732752440.000000
#28 e^9=-380414021471186820.000000 sum=-344484119037083140.000000 error=380414021471186820.000000
#29 e^9=-3793858754672106000.000000 sum=-4138342873709189100.000000 error=3793858754672106000.000000
#30 e^9=30079880126328840000.000000 sum=25941537252619649000.000000 error=30079880126328840000.000000
#31 e^9=516827031261468300000.000000 sum=542768568514087940000.000000 error=516827031261468300000.000000
#32 e^9=-1598933627965167400000.000000 sum=-1056185059451079400000.000000 error=1598933627965167400000.000000
#33 e^9=-14390402651686506000000.000000 sum=-1544656771137585000000.000000 error=14390402651686506000000.000000
#34 e^9=1.#INF00 sum=1.#INF00 error=1.#INF00
```

2번과 4번 문제처럼 항을 하나씩 더해서 얻은 결과

```
C:\Users\jack8\Desktop\코딩\수치해석 과제\HW#H...
참값 = 8103.083928
# 1 e^9=1.000000 error=1.000000
# 2 e^9=10.000000 error=9.000000
# 3 e^9=50.500000 error=40.500000
# 4 e^9=172.000000 error=121.500000
# 5 e^9=445.375000 error=273.375000
# 6 e^9=937.450000 error=492.075000
# 7 e^9=1675.562500 error=738.112500
# 8 e^9=2624.564286 error=949.001786
# 9 e^9=3692.191295 error=1067.627009
#10 e^9=4759.818304 error=1067.627009
#11 e^9=5720.682612 error=960.864308
#12 e^9=6506.844318 error=786.161707
#13 e^9=7096.465598 error=589.621280
#14 e^9=7504.664946 error=408.199348
#15 e^9=7767.078812 error=262.413866
#16 e^9=7924.527132 error=157.448320
#17 e^9=8013.091812 error=88.564680
#18 e^9=8059.978995 error=46.887183
#19 e^9=8083.422587 error=23.443592
#20 e^9=8094.527446 error=11.104859
#21 e^9=8099.524633 error=4.997187
#22 e^9=8101.666284 error=2.141651
#23 e^9=8102.542414 error=0.876130
#24 e^9=8102.885248 error=0.342834
#25 e^9=8103.013811 error=0.128563
#26 e^9=8103.060093 error=0.046283
#27 e^9=8103.076114 error=0.016021
#28 e^9=8103.081454 error=0.005340
#29 e^9=8103.083171 error=0.001717
#30 e^9=8103.083703 error=0.000533
#31 e^9=8103.083863 error=0.000160
#32 e^9=8103.083910 error=0.000046
#33 e^9=8103.083923 error=0.000013
#34 e^9=8103.083926 error=0.000004

Process exited after 0.744 seconds with return value 0
계속하려면 아무 키나 누르십시오 . . .
```

```
C:\Users\jack8\Desktop\코딩\수치해석 과제\HW#H...
참값 = 0.000123
# 1 e^-9=1.000000 error=1.000000
# 2 e^-9=-8.000000 error=-9.000000
# 3 e^-9=32.500000 error=40.500000
# 4 e^-9=-89.000000 error=-121.500000
# 5 e^-9=184.375000 error=273.375000
# 6 e^-9=-307.700000 error=-492.075000
# 7 e^-9=430.412500 error=738.112500
# 8 e^-9=-518.589286 error=-949.001786
# 9 e^-9=549.037723 error=1067.627009
#10 e^-9=-518.589286 error=-1067.627009
#11 e^-9=442.275022 error=960.864308
#12 e^-9=-343.886684 error=-786.161707
#13 e^-9=245.734596 error=589.621280
#14 e^-9=-162.464752 error=-408.199348
#15 e^-9=99.949114 error=262.413866
#16 e^-9=-57.499205 error=-157.448320
#17 e^-9=31.065474 error=88.564680
#18 e^-9=-15.821709 error=-46.887183
#19 e^-9=7.621883 error=23.443592
#20 e^-9=-3.482977 error=-11.104859
#21 e^-9=1.514210 error=4.997187
#22 e^-9=-0.627441 error=-2.141651
#23 e^-9=0.248689 error=0.876130
#24 e^-9=-0.094145 error=-0.342834
#25 e^-9=0.034418 error=0.128563
#26 e^-9=-0.011865 error=-0.046283
#27 e^-9=0.004156 error=0.016021
#28 e^-9=-0.001184 error=-0.005340
#29 e^-9=0.000532 error=0.001717
#30 e^-9=-0.000000 error=-0.000533
#31 e^-9=0.000160 error=0.000160
#32 e^-9=0.000113 error=0.000046
#33 e^-9=0.000126 error=0.000013
#34 e^-9=0.000123 error=0.000004

Process exited after 0.7205 seconds with return value 0
계속하려면 아무 키나 누르십시오 . . .
```

항을 (x/n)을 곱해가며 만들어서 얻은 결과

e^9 의 참값=8103.08, 계산값=8103.083926 e^{-9} 의 참값= 1.23409×10^{-4} 계산값=0.000123