**고급소프트웨어 실습 4주차 과제**

20171666 이예은

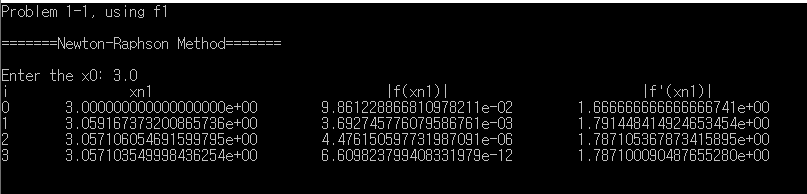
**프로그램 구동 방법**

* 해당 프로그램은 다음 차례대로 알맞은 초기값을 받고 그에 따른 결과를 보여준다.
* f1 함수를 사용한 newton-raphson method와 secant method. (실습 1-1)
* f2 함수를 사용한 newton-raphson method와 secant method. (실습 1-1)
* f3 함수와 newton-raphson method를 활용해 총 4개의 근 산출. (실습 1-2)
* f=lnx − 1 = 0 함수와 newton-raphson method를 사용해 double일 때 결과. (실습 1-4)
* f=lnx − 1 = 0 함수와 newton-raphson method를 사용해 float일 때 결과. (실습 1-4)
* f1 함수를 사용한 bisection method. (과제 1)
* f2 함수를 사용한 bisection method. (과제 1)
* f3 함수를 사용한 bisection method. (과제 1)
* f(α) = Asinα cosα +Bsin2α −Ccosα −E sinα = 0 함수를 사용해 각도 산출. (과제 2)

**실습 문제 1-1**

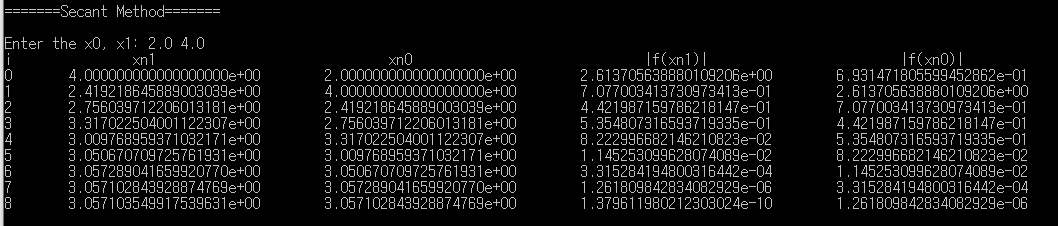
1. 두 방법에 대한 초기값을 각각 x0 = 3.0과 x0 = 2.0, x1 = 4.0으로 설정하여 자신이 작성한 프로그램을 수행시켜 자신의 프로그램이 원하는 근을 정확히 찾고 있는지 분석하라. 과연 근이 맞는지 어떻게 확인할 수 있을까? 논리적으로 타당한 방법으로 분석한 내용을 보고서에 기술하라.

* x0=3.0을 넣어 newton’s 방법을 실행했을 때 결과는 다음과 같다.



여기서 |f(xn1)|이 마지막엔 약 6.6\*10^-12로 0에 매우 가까워진다. 그러므로 종료조건인 0.000001 보다 차이가 작아져, 우리는 해당 식의 근이 약 3.057이란 것을 알 수 있다.

x0 = 2.0, x1 = 4.0을 넣어 scent 방법을 실행했을 때 결과는 다음과 같다.



여기서도 |f(xn1)|이 마지막엔 약 1.37\*10^-10으로 0에 매우 가까워진다. 이 때도 실제 근과 차이가 0.000001보다 작아지기 때문에 해당 식의 근을 약 3.057로 구할 수 있다.

1. 위에서 산출한 결과를 볼 때, 각 방법의 근의 수렴 속도가 과연 앞에서 설명한 속도로 수렴하는지 비교 분석한 후 그 결과를 보고서에 기술하라.

Newton’s 방법에서의 절대오차를 다 구해보면,

|  |  |
| --- | --- |
| ε0 | 0.057 |
| ε1 | 0.002 |
| ε2 | 0.00003 |

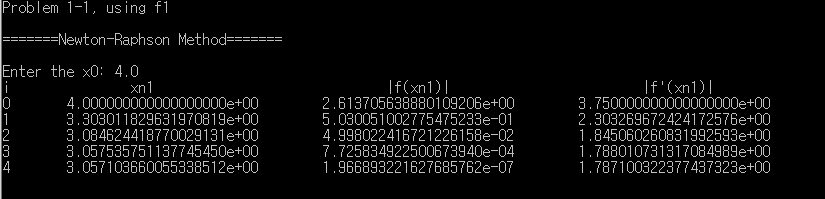
이 된다. 이 경우에는 근까지 도달하는데 걸리는 시간이 매우 짧아 설명하는 속도로 수렴하는지 확인이 쉽지 않다.

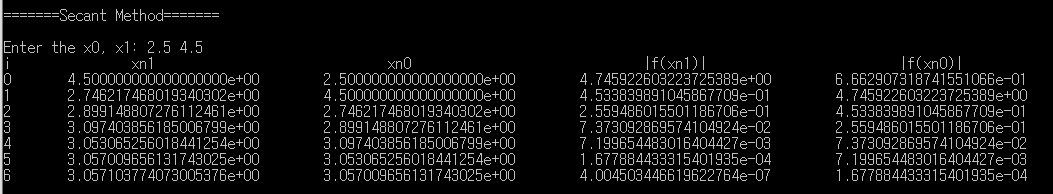
Secant 방법의 절대오차를 구해보면,

|  |  |
| --- | --- |
| ε0 | 0.043 |
| ε1 | 0.638 |
| ε2 | 0.301 |
| ε3 | 0.26 |
| ε4 | 0.048 |
| ε5 | 0.007 |
| ε6 | 0.0001 |
| ε7 | 0.000001 |

이므로 이론과 맞게 수렴하는 것을 볼 수 있다.

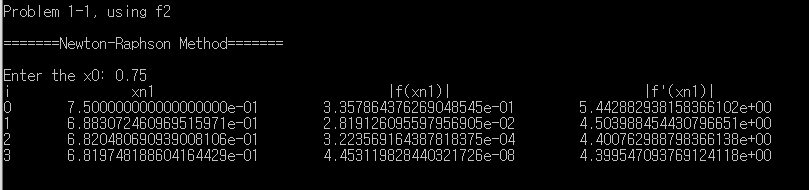
1. 위에서 제시한 초기값 외에 임의의 초기값들을 사용하여 자신이 작성한 프로그램을 수행한 후, 이 두방법이 항상 임의의 초기값에 대해 빠르게 수렴하는지 보고서에 기술하라.

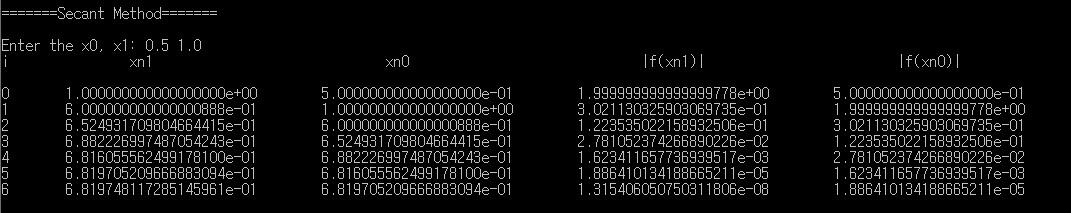




* 다음 결과와 같이 적절한 초기 값을 주면 빠르게 수렴한다. 하지만 잘못된 초기값을 줄 경우, 발산하거나 제대로 된 값을 찾지 못하는 경우가 생긴다. 그러므로 항상 임의의 초기값에 대해 빠르게 수렴하지는 않는다.

1. 위에서 작성한 프로그램을 사용하여 함수 f2(x) = x+1−2sinπx = 0에 대해 초기값을 각각 x0 = 0.75과 x0 = 0.5, x1 = 1.0으로 설정하여 위 문제를 반복하라.





* 각각 마지막 |f(xn1)|이 4.45\*10^-8, 1.31\*10^-8로 0에 매우 가까워진다. 그러므로 해당 식의 근을 약 6.81\*10^-10로 근사할 수 있다.

Newton’s 방법의 절대오차를 구해보면,

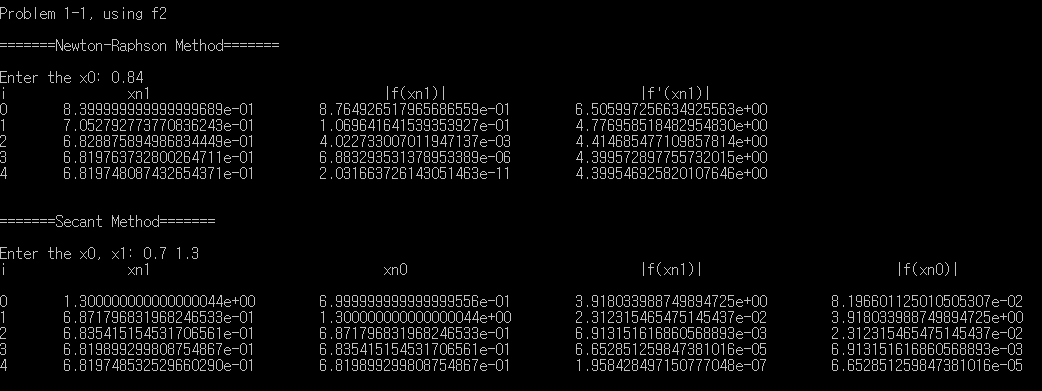
|  |  |
| --- | --- |
| ε0 | 0.069 |
| ε1 | 0.007 |
| ε2 | 0.001 |

secant 방법의 절대오차를 구해보면,

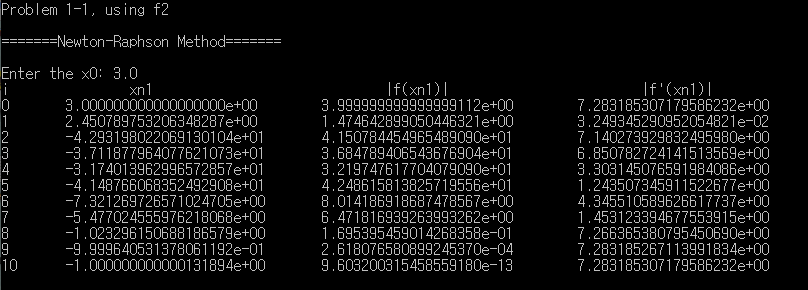
|  |  |
| --- | --- |
| ε0 | 0.319 |
| ε1 | 0.081 |
| ε2 | 0.031 |
| ε3 | 0.007 |
| ε4 | 0.0003 |
| ε5 | 0.000004 |

으로 newton’s의 경우 구하는데 걸린 시간이 짧아 제대로 확인이 힘들지만 두 방법 모두 대체로 이론대로 수렴하는 것을 확인할 수 있다.

임의로 x0 = 0.84와 x0 = 0.7, x1 = 1.3로 했을 때 결과는 다음과 같이 잘 나오는 것을 볼 수 있다.



하지만 x0=3.0으로 잘못된 초기값을 줄 경우, 제대로 된 근을 못 찾는 것을 확인할 수 있다. 그러므로 모든 임의 초기값에 대해 빠르게 수렴하지는 않는다.



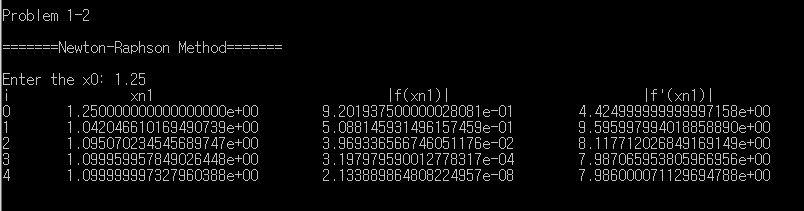
**실습문제 1-2**

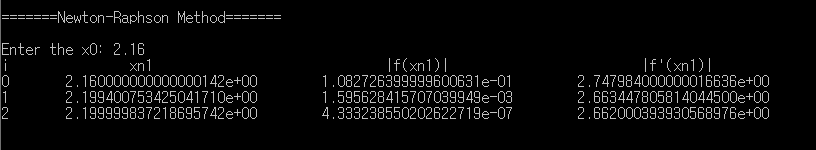
1. 다음과 같은 다항식에 대한 방정식을 고려하자.

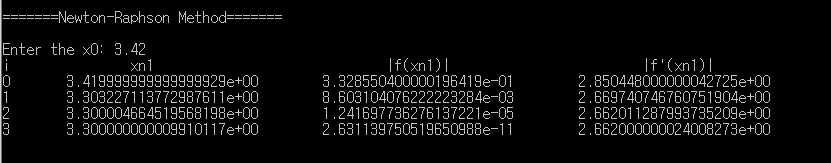
f3(x) = x^4 −11.0x^3 +42.35x^2 −66.55x+35.1384 = 0

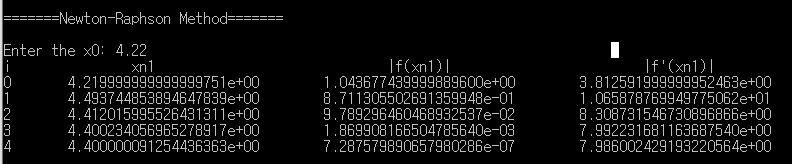
지금 어떤 근 분리 (root separation) 방법을 사용하여, 이 방정식은 네 개의 서로 다른 실근을 가지며, 각 실근은 [1.02,1.48], [1.95,2.37], [3.11,3.73], [3.83,4.61] 구간에 존재한다는 사실을 밝혀냈다. 이 사실을 근거로 Newton-Raphson 방법을 사용하여 모든 실근을 구하라.

* Newton’s 방법을 사용하기 위해 초기값을 각 실근 구간의 중간값으로 넣어주었다.







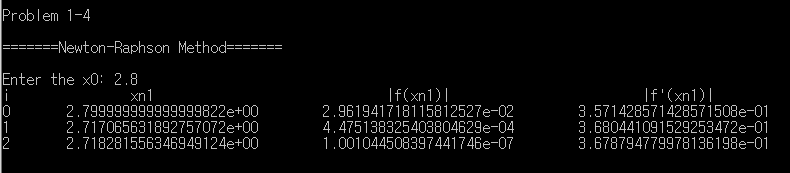


그 결과로 모두 |f(xn1)|이 0에 매우 가까워 근으로 근사할 수 있는 값들이 산출되었다. 각각 약 1.09, 2.19, 3.3, 4.4가 근이 된다.

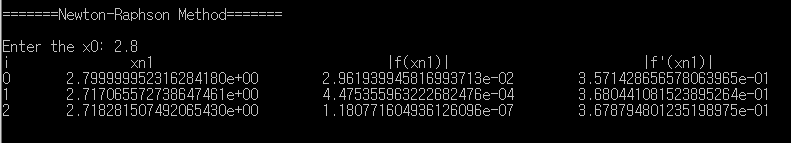
**실습문제 1-4**

1. 다음 f1(x) = lnx − 1 = 0과 같이 근을 알고 있는 비선형 방정식을 고려하 자 (잘 알다시피 근은 e = 2.718281828459045235360287471352···임). 이 방 정식의 근을 Newton-Raphson 방법을 사용하여 구하려 하는데, 적절한 초기 값 x0에 대해 자신이 작성한 double-precision 버전과 single-precision 버전 각각을 사용하여 근을 구하여 보자. 이때 부동 소수점 연산의 정밀도가 다 른 두 방법이 구한 근의 값이 정확한 근 e와 비교하여 어떤 차이가 있는지, 자신이 알아낸 사실을 보고서에 상세히 기술하라.

* 먼저 double로 구했을 경우 값은 다음과 같이 근사된다.



Float로 근사했을 때는 다음과 같다.

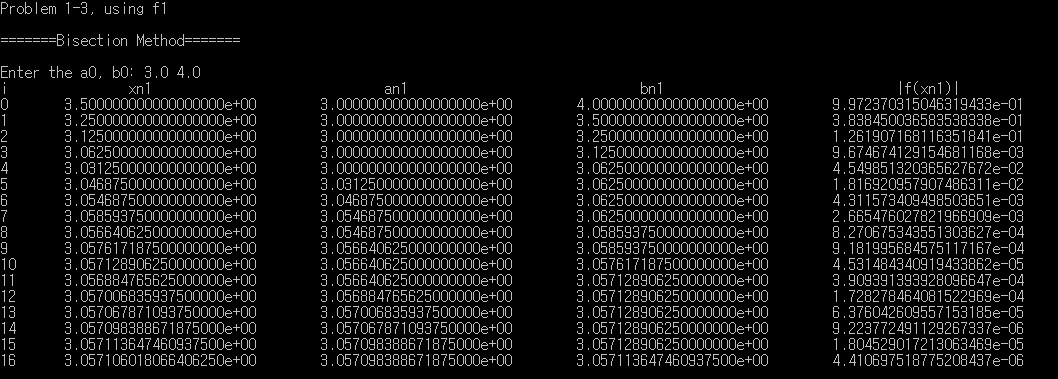


두 경우 모두 값은 원래 근인 2.718에 매우 가까워지는 것을 확인할 수 있다. double의 경우 원래 근과는 약 0.0000002715정도 차이가 나지만, float의 경우 0.0000003209정도 차이가 난다. 즉, double로 계산했을 경우가 조금 더 정확하다.

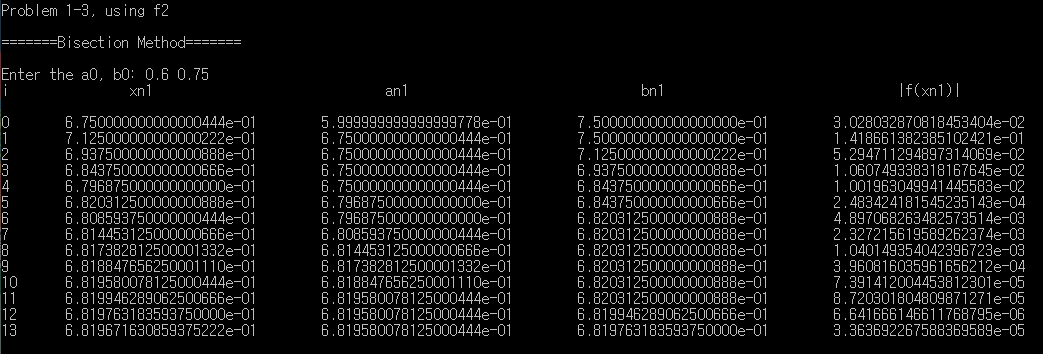
**과제 1**

1. 실습 시간에 사용한 세 개의 함수 f1(x), f2(x), 그리 고 f3(x)에 대해 적절한 초기 구간을 사용하여 올바르게 근에 수렴하는지에 대한 분석 내용을 보고서에 기술하라.

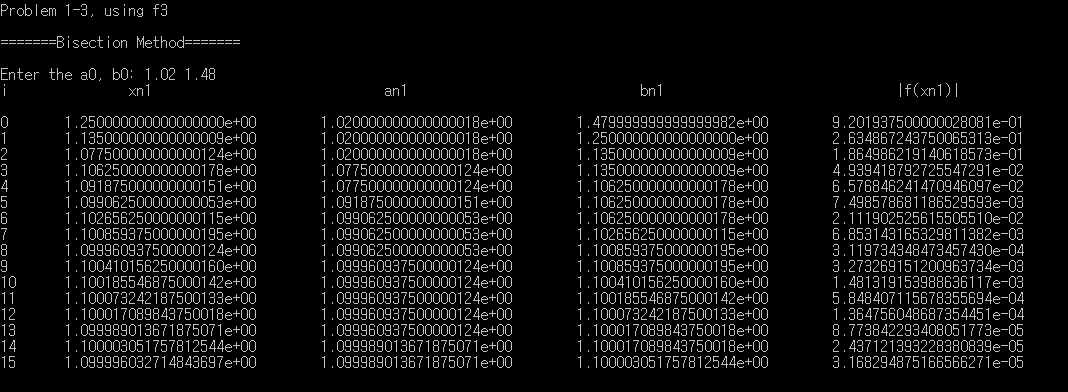
* f1(x)에 대한 값은 다음과 같다.



f2(x)에 대한 값은 다음과 같다.



f3(x)에 대한 값은 다음과 같다.



위의 결과들을 보면, 각각 근이 3.057, 6.81\*10^-1, 1.09로 실습 1-1, 1-2에서 구한 값들과 매우 근사하다는 것을 알 수 있다. 또한 이 방법으로 구한 근들의 |f(xn1)| 또한 0에 매우 가까우므로 성공적으로 근을 근사했음을 알 수 있다.

1. Bisection 방법의 수렴 속도는 선형적인, 즉 εn+1 ≈ 1/2 εn 형태를 보인다. 위의 세 함수에 대한 방정식을 대상으로 Newtown-Raphson 방법, Secant 방법, 그리고 Bisection 방법을 적용해보고, 과연 각 방법이 이론적인 수렴 속도를 보이는지를 분석하고 그 내용을 보고서에 기술하라.

위에 다른 방법은 이미 속도를 구했으므로 bisection 방법만 구해보면 다음과 같다.

함수 f1일 때,

|  |  |
| --- | --- |
| ε0 | 0.4429 |
| ε1 | 0.193 |
| ε2 | 0.068 |
| ε3 | 0.0055 |
| ε4 | 0.026 |
| ε5 | 0.011 |
| ε6 | 0.003 |
| ε7 | 0.001 |
| ε8 | 0.001 |
| ε9 | 0.0005 |
| ε10 | 0.00002 |
| ε11 | 0.001 |
| ε12 | 0.0001 |
| ε13 | 0.00004 |
| ε14 | 0.00001 |
| ε15 | 0.00001 |

함수 f2일 때,

|  |  |
| --- | --- |
| ε0 | 0.0069 |
| ε1 | 0.03054 |
| ε2 | 0.01174 |
| ε3 | 0.00234 |
| ε4 | 0.00236 |
| ε5 | 0.00004 |
| ε6 | 0.00116 |
| ε7 | 0.00056 |
| ε8 | 0.00026 |
| ε9 | 0.00016 |
| ε10 | 0.00001 |
| ε11 | 0.00003 |
| ε12 | 0.00001 |

함수 f3일 때,

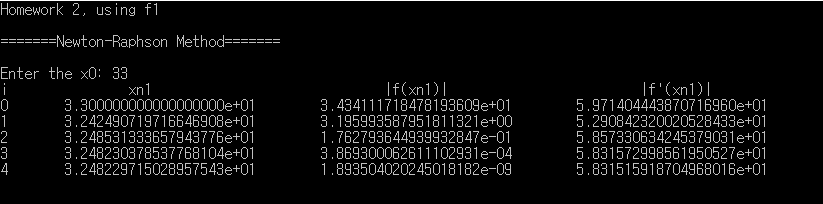
|  |  |
| --- | --- |
| ε0 | 0.15001 |
| ε1 | 0.035 |
| ε2 | 0.0224 |
| ε3 | 0.0062 |
| ε4 | 0.00819 |
| ε5 | 0.00093 |
| ε6 | 0.00201 |
| ε7 | 0.00081 |
| ε8 | 0.00003 |
| ε9 | 0.00001 |
| ε10 | 0.00019 |
| ε11 | 0.000083 |
| ε12 | 0.000027 |
| ε13 | 0.000001 |
| ε14 | 0.000013 |

위 결과들을 쭉 살펴보면, 오차가 존재하지만 대체로 이론 값인 1/2 만큼씩 절대오차가 줄어드는 것을 확인할 수 있다.

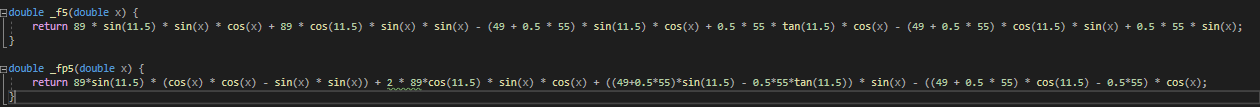
**과제 2**

1. 이 그림에 표시된 여러 인자에 대하여 다음과 같은 방정식이 성립한다고 하는데, f(α) = Asinα cosα +Bsin2α −Ccosα −E sinα = 0, 여기서 각 상수 값은 다음과 같이 정의된다. A = lsinβ1, B = l cosβ1, C = (h+0.5D)sinβ1 −0.5Dtanβ1, E = (h+0.5D) cosβ1 −0.5D. 만약 l = 89 인치, h = 49 인치, D = 55 인치, 그리고 β1 = 11.5도이라면, 각도 α는 대략 33도 정도가 됨이 알려져 있다. 여러분이 작성한 Newton-Raphson 방법을 사용하여 정확한 각도 α 값을 구하라.

* Newton’s 방법의 초기값을 33, f(α)를 적절히 구현해 결과를 보면 다음과 같다.



실습 1-1과 같은 newton’s 방법을 사용했으며, 사용한 f 식은 다음과 같다.



각각에 맞는 값을 넣어 구성하였다. 이 f 식을 사용해 newton’s 방법을 구현한 함수를 부르면 위와 같은 결과가 나오며, |f(xn1)|이 1.8\*10^-9로 0에 한없이 가깝다. 그러므로 그 때의 xn1값 32.4를 각도 α라 할 수 있다.