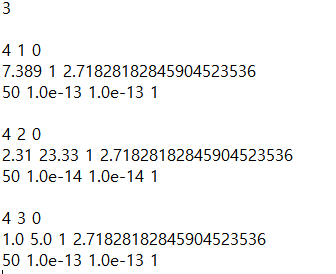
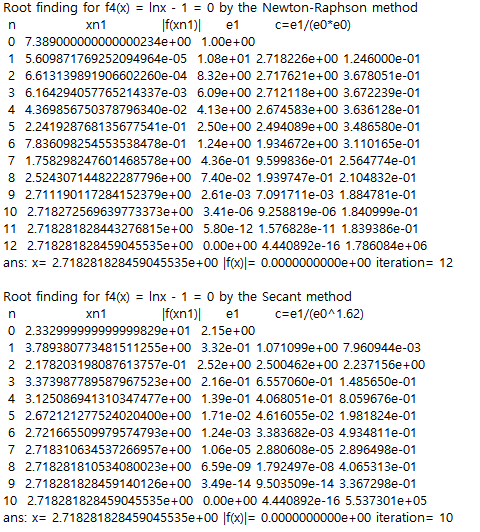
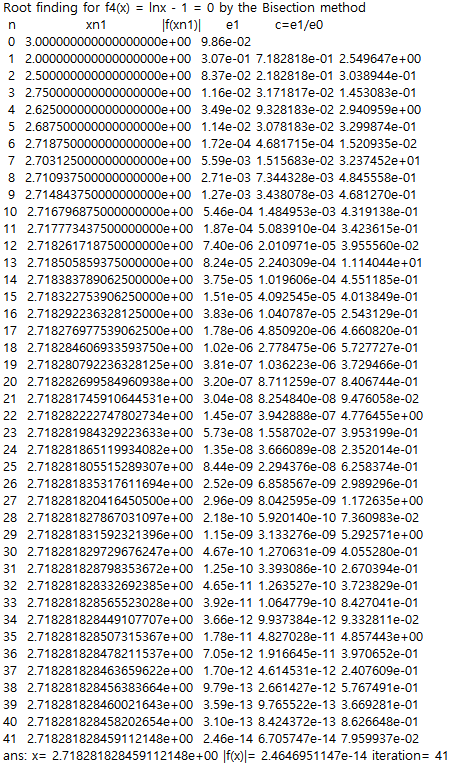
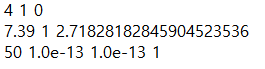
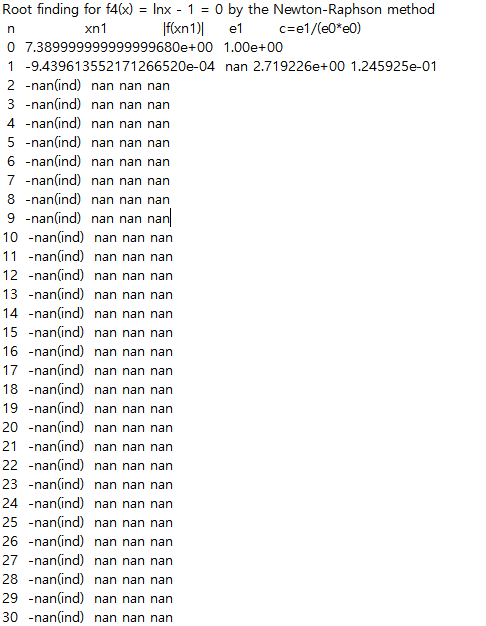
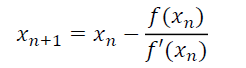
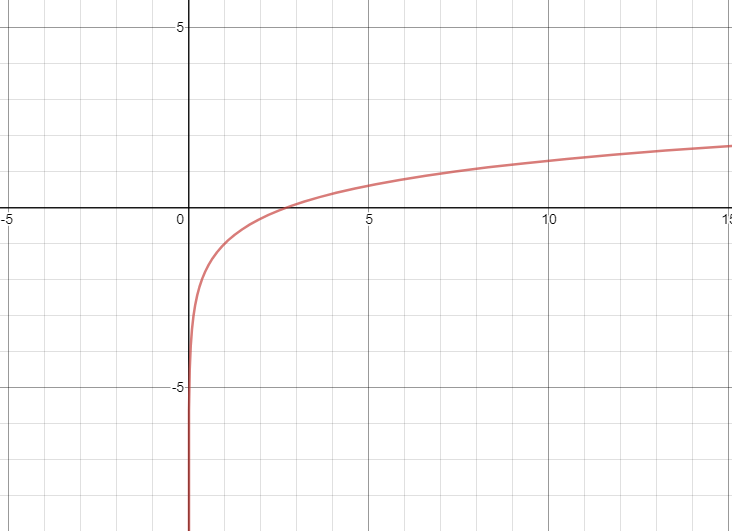
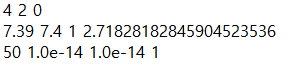
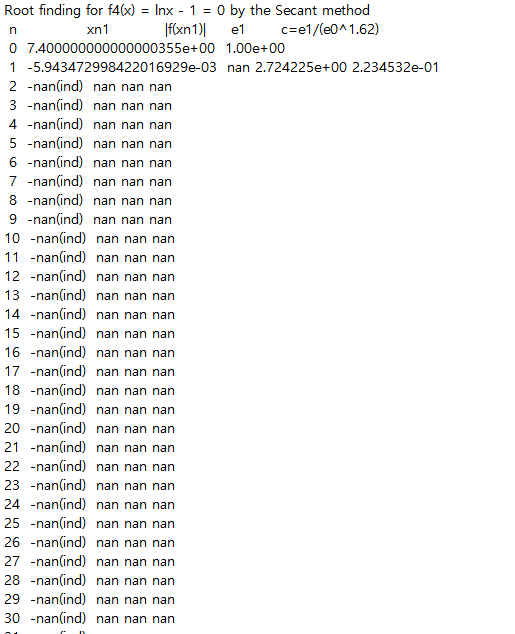
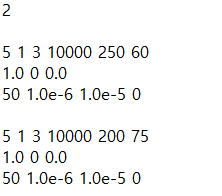
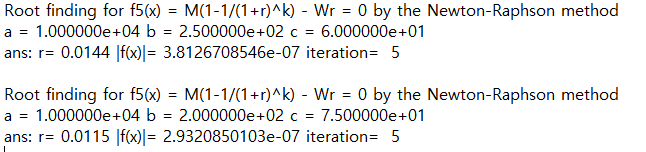
**고급소프트웨어실습1 [CSE4152-04]**

4주차 숙제 : Nonlinear Equation Root Finding

**학번 : 20171640**

**이름 : 박수진**

1. Problem 4
   1. 입력 및 출력
      1. 입력  
         
         1. Newton-Raphson method  
            초기값 : 7.389  
            maxIter : 50  
            delta : 1.0e-13  
            epsilon : 1.03-13
         2. Secant method  
            초기값 : 2.31 / 23.33  
            maxIter : 50  
            delta : 1.0e-13  
            epsilon : 1.03-13
         3. Bisection method  
            초기값 : 1.0 / 5.0  
            maxIter : 50  
            delta : 1.0e-13  
            epsilon : 1.03-13
      2. 출력  
           
           
          
   2. 분석
      1. Newton-Raphson method   
         - Convergence Rate : root finding 직전에 (iteration이 12일 때) convergence rate가 1.786084e+06으로 매우 빠르다.  
         - 반복 횟수 : 12번으로 초기값 설정이 Secant method와 달라서 반복횟수가 Secant method와 비슷하다. 하지만 통상적으로 Newton-Secant method가 Secant method보다 빠르다.  
         - 구한 해의 정확도 : 마지막 e1의 값이 4.440892e-16으로 bisection method에 비해 매우 작다. 또한 root finding한 x를 f{x}에 대입했을 때 값이 0으로 매우 정확한 편이다.
      2. Secant method  
         - Convergence Rate : root finding 직전에 (iteration이 10일 때) convergence rate가 5.537301e+05으로 매우 빠르다.  
         - 반복 횟수 : 10번으로 초기값 설정이 Newton-Raphson method와 달라서 반복횟수가 Newton-Raphson method와 비슷하다. 하지만 통상적으로 Newton-Secant method가 Secant method보다 빠르다.  
         - 구한 해의 정확도 : 마지막 e1의 값이 4.440892e-16으로 bisection method에 비해 매우 작다. 또한 root finding한 x를 f{x}에 대입했을 때 값이 0으로 매우 정확한 편이다.
      3. Bisection method  
         - Convergence Rate : root finding 직전에도 convergence rate가 빨라지지 않는다. 모든 iteration에서 convergence rate가 급격한 변화 없이 비슷하다.  
         - 반복 횟수 : 41회로 Newton-Raphson과 Secant method에 비해 엄청 크다. 즉 반복 횟수가 위 두 방법보다 많아서 finding하는데 시간이 오래 걸린다.  
         - 구한 해의 정확도 : 마지막 e1의 값이 6.705747e-14로 Newton-Raphson method와 Secant method에 비해서 크다. 또한 root finding한 x를 f(x)에 대입했을 때 절댓값이 2.4646951147e-14로 위의 두 방법에 비해 정확도가 떨어진다.
   3. 발산하는 경우
      1. Newton-Raphson method  
           
           
           
         위와 같이 초기값을 7.39로 입력하면 아래와 같은 결과가 나타난다.  
           
           
           
         Newton – Raphson method의 식은 아래와 같다.  
           
           
         출력 값 중 첫번째 값(n = 1)을 보면 x값이 음수가 나오는 것을 볼 수 있다. 이는 x0에서의 미분 값(즉, x0에서 접선의 기울기)가 매우 작아서 x0에서 주어진 범위보다 큰 값을 빼서 그 다음 x값(x1)이 음수가 나온 것이다. ln(x) – 1의 정의역은 (0, inf]인데 -9.439613… 값은 그 범위를 벗어나기 때문에 해당하는 f(x)값이 존재하지 않는다.  
           
         ln(x) – 1은 정의역 내에서 x값이 커질수록 미분 값은 계속 작아진다. 그러나 미분 값이 0이 되는 경우는 없다. 그렇기 때문에 x 값이 수렴하지 않는 경우는 2가지이다.  
           
         1) 초기값이 정의역을 벗어나는 경우 (즉, 0보다 작거나 같은 경우)  
         2) f’(x0) 값이 매우 작아서(즉, f(x0)/f’(x0)이 매우 커서) x0에서 f(x0)/f’(x0)을 뺀 값이 정의역을 벗어나는 경우 (실험적으로 7.39 이상의 초기값을 입력 시 이와 같은 결과가 나온다.)  
           
         \* ln(x) - 1 의 그래프는 아래와 같다.  
         
      2. Secant method  
           
           
         위와 같이 초기값을 7.39 / 7.4로 입력하면 아래와 같은 결과가 나타난다.  
           
           
           
           
           
         Secant method의 식은 다음과 같다.  
           
           
         여기서 f(x1) \* ((x1 – x0) / (f(x1) – f(x0)) 은 x1에서의 함수 값과 x0와 x1사이의 기울기의 역수를 곱한 것과 같다. 정의역 내에서 x1에서의 함수 값이 무한대로 발산하는 경우는 없으므로, 발산하는 경우는 (x0와 x1이 같지 않다면) x0와 x1사이의 기울기가 매우 작아서 역수를 곱했을 때 f(x1) \* ((x1 – x0) / (f(x1) – f(x0))의 전체 값이 매우 커져서 x1에서 뺐을 때 정의역 밖의(0보다 작거나 같은) x값이 나오는 경우이다.  
           
         Secant method에서 발산하는 경우는 아래와 같다.  
           
         1) 초기값 중 적어도 하나가 정의역을 벗어나는 경우 (즉, 0보다 작거나 같은 경우)  
         2) x0와 x1 사이의 평균 변화량 값이 매우 작아서(즉, f(x1) \* ((x1 – x0) / (f(x1) – f(x0)) 값이 매우 커서) x1에서 f(x1) \* ((x1 – x0) / (f(x1) – f(x0))을 뺀 값이 정의역을 벗어나는 경우  
         3) 두 초기값 x0와 x1이 같을 경우 (평균 변화율이 0이므로 역수를 취했을 때 무한대로 발산한다.)  
           
         \* Newton-Raphson method의 경우 x0에서의 순간 변화율이 작을 때 발산하는데, lnx – 1은 순간 변화율이 계속 감소하므로 일정한 값(실험적으로 약 7.39)부터 계속 발산한다. 그러나 Secant method의 경우 x1과 x0의 평균 변화율이 작을 때 발산하므로, 그 두 점을 특정할 수 없다. 위에서 초기값이 7.39 / 7.4 일 때 발산함을 보였지만 1.0 / 18.0 등의 특정할 수 없는 평균 변화율이 작은 두 점에서도 발산한다.
2. Problem 5
   1. 입력 및 출력
      1. 입력  
           
         1. W = 10000, M = 250, k = 60일 경우  
            초기값 : 1.0  
            maxIter : 50  
            delta : 1.0e-6  
            epsilon : 1.03-5
         2. W = 10000, M = 200, k = 75일 경우  
            초기값 : 1.0  
            maxIter : 50  
            delta : 1.0e-6  
            epsilon : 1.03-5  
              
            \* 두 경우 모두 Newton-Raphson method를 이용하고 convergence check 하지 않고 중간 결과 출력하지 않았다.
      2. 출력  
         
         1. W = 10000, M = 250, k = 60일 경우  
            r : 0.0144
         2. W = 10000, M = 200, k = 75일 경우  
            r : 0.0115
      3. 유리한 상환 방법  
         2번 방법이 1번 방법보다 이자율이 낮으므로 2번 방법이 더 유리한 상환 방법이다.