



7. 2 비선형 방정식의 해

7.0 서론

7.1 고정점 반복법

7.2 Newton-Raphson 법

7.3 이분법

7.4 가위치법



7.0 서론

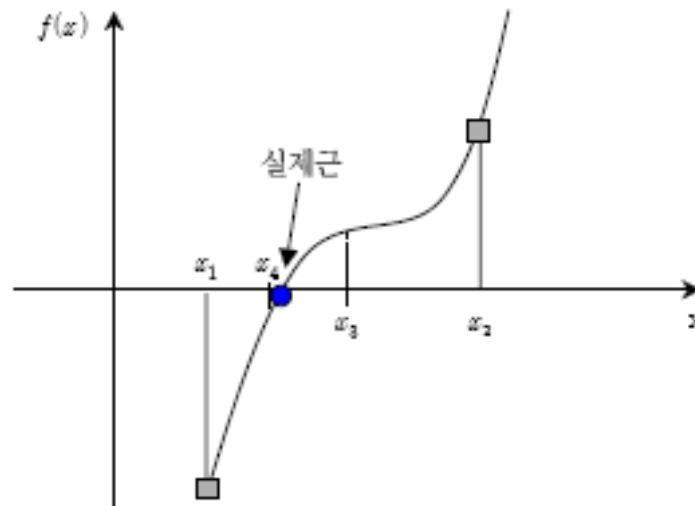
- 방정식의 근을 찾는 방법
 - 근의 공식을 이용한 해석적인 방법
 - 근사해
 - 그래프 이용법
 - 함수식을 그래프로 그려 x 축과 만나는 점, 즉 $f(x)=0$ 이 되는 근 x 를 찾음
 - 정확도가 떨어짐
 - 시행 착오법(trial and error)
 - 많은 임의의 수를 대입하여 그 중 조건에 맞는 값을 근으로 취함
 - 비효율적이고 계산 시간이 오래 걸림
 - 구간법 및 개방법
 - 개방법 : 근을 포함하는 구간의 양 끝이 아닌 한 개 이상의 초기값을 이용
 - 고정점 반복법, Newton-Raphson법, 할선법
 - 구간법 : 근을 포함하는 구간의 양 끝을 초기값으로 이용
 - 이분법, 가위치법

■ 반복 종료를 위한 임계값 결정

$$E_r = \frac{x^{now} - x^{old}}{x^{now}} \times 100 \%$$

x^{now} : 현재의 반복 계산에서 구한 근

x^{old} : 이전의 반복 계산에서 구했던 근



7.1 고정점 반복법

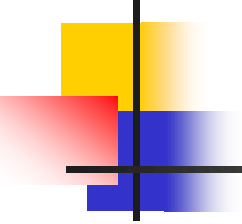
(Fixed-point iteration)

- 방정식의 형태를 변형하여 근을 구하는 방법

$$x = g(x)$$

- 이전 계산 $g(x)$ 에서 얻은 x 의 값을 이용하여 새로운 x 값을 예측

$$x_{i+1} = g(x_i)$$

- 
- 예제) 고정점 반복법을 이용하여 $f(x) = x - e^{-x}$ 의 근을 구하라.

풀이. 주어진 함수를 변형하면 $x_{i+1} = e^{-x}$ 이다.

따라서 초기값 $x_0 = 0$ 을 사용하여 2회 반복 계산하면 다음과 같다.

$$x_1 = e^0 = 1$$

$$x_2 = e^1 = 0.3679$$

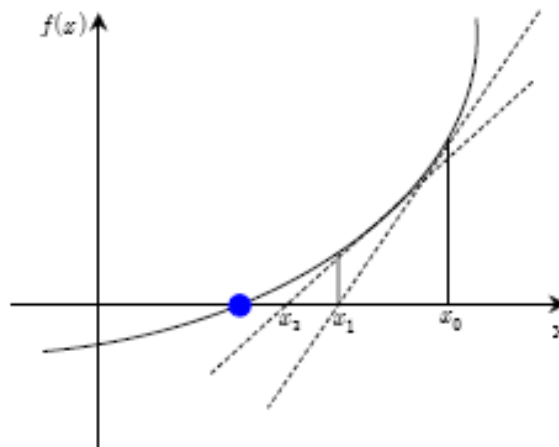


상대오차가 0.0001이 될 때까지 반복한 결과 실제값인 0.56714329에 수렴함을 알 수 있음

i (반복 횟수)	x_i	상대 오차(%)
0	0.0000	
1	1.0000	100
2	0.3679	171.828
3	0.6922	46.856
4	0.5005	38.309
5	0.6062	17.447
6	0.5454	11.157
7	0.5796	5.903
8	0.5601	3.481
9	0.5711	1.931
10	0.5649	1.098

7.2 Newton-Raphson법

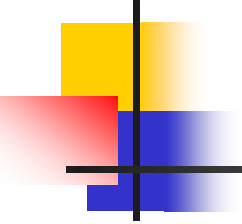
- 근을 구하기 위한 가장 효율적인 방법
 - 초기값 x_0 이 주어졌을 때 점 $(x_0, f(x_0))$ 에 접하는 접선을 구하고, 이 접선이 x 축과 만나는 점이 새로운 근 x 가 됨



| Newton-Raphson법의 원리

이분은 특정 지점에서 나타나는
접선의 기울기를 뜻한다

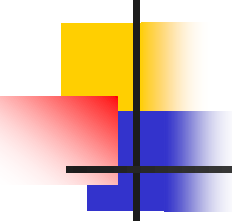
$$x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)}$$

- 
- 예제) Newton-Raphson법을 사용해서 $f(x) = \log(x + 5.0) + x$ 의 근을 구하라. 초기값은 $x_0 = 7.0$ 으로 한다.

풀이. $f(x) = \log(x + 5.0) + x$ 이고 $f'(x) = \frac{1}{x + 5.0} + 1$ 이므로

$$x_1 = 7 - \frac{8.0792}{1.0833} = -0.4580 \quad \text{이 되며,}$$

$$x_1 = -0.4580 - \frac{0.1992}{1.2202} = -0.6213 \quad \text{이 된다.}$$



상대오차가 0.0001이 될 때까지 반복한 결과 고정점
반복법보다 훨씬 더 빨리 근에 수렴함을 알 수 있음

i(반복 횟수)	x_i	상대 오차(%)
0	7.0000	
1	-0.4577	1629.4000
2	-0.6213	26.3270
3	-0.6376	2.5640
4	-0.6393	0.2684
5	-0.6395	0.0283

MATLAB Code Example

```
% Experiment 4
% Newton Raphson Method
clear all
close all
clc

% Change here for different functions
f=@(x) cos(x)-3*x+1
%this is the derivative of the above function
df=@(x) -sin(x)-3
% Change lower limit 'a' and upper limit 'b'
a=0; b=1;
x=a;
for i=1:1:100
    x1=x-(f(x)/df(x));
    x=x1;
end
sol=x;
fprintf('Approximate Root is %.15f',sol)

a=0;b=1;
x=a;
er(5)=0;
for i=1:1:5
    x1=x-(f(x)/df(x));
    x=x1;
    er(i)=x1-sol;
end
plot(er)
xlabel('Number of iterations')
ylabel('Error')
title('Error Vs. Number of iterations')

f =
    @(x)cos(x)-3*x+1

df =
    @(x)-sin(x)-3

Approximate Root is 0.607101648103123
```

Python Code Example

```
def derivative(f, x, h):
    return (f(x+h) - f(x-h)) / (2.0*h) # might want to return a small non-zero if ==0

def quadratic(x):
    return 2*x*x-5*x+1 # just a function to show it works

def solve(f, x0, h):
    lastX = x0
    nextX = lastX + 10* h # "different than lastX so loop starts OK
    while (abs(lastX - nextX) > h): # this is how you terminate the loop - note use of abs(
        newY = f(nextX) # just for debug... see what happens
        print "f(", nextX, ") = ", newY # print out progress... again just debug
        lastX = nextX
        nextX = lastX - newY / derivative(f, lastX, h) # update estimate using N-R
    return nextX

xFound = solve(quadratic, 5, 0.01) # call the solver
print "solution: x = ", xFound # print the result
```

output:

```
f( 5.1 ) = 27.52
f( 3.31298701299 ) = 6.38683083151
f( 2.53900845771 ) = 1.19808560807
f( 2.30664271935 ) = 0.107987672721
f( 2.28109300639 ) = 0.00130557566462
solution: x = 2.28077645501
```

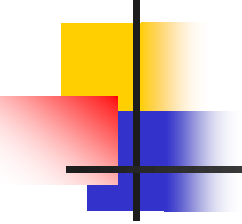
7.3 이분법

(Bisection method)

- 중간값 정리(intermedia value theorem)을 토대
 - 함수 $f(x)$ 가 구간 (x_1, x_2) 에서 연속이고, $f(x_1)$ 과 $f(x_2)$ 의 부호가 서로 반대이면 x_1 과 x_2 사이에 적어도 한 개의 근이 존재한다는 원리
 - 이때 두 점 사이를 다시 이등분한 점 x_3 을 근으로 가정

$$x_3 = \frac{x_1 + x_2}{2}$$

- if $f(x_1)f(x_3) < 0$ 이면 근은 x_1 과 x_3 사이에 존재
else 근은 x_2 과 x_3 사이에 존재
- 이를 다시 이등분 반복하여 x_4, \dots, x_n 을 구해 가다가 임계값에 도달시 이때의 x_m 을 근으로 추정

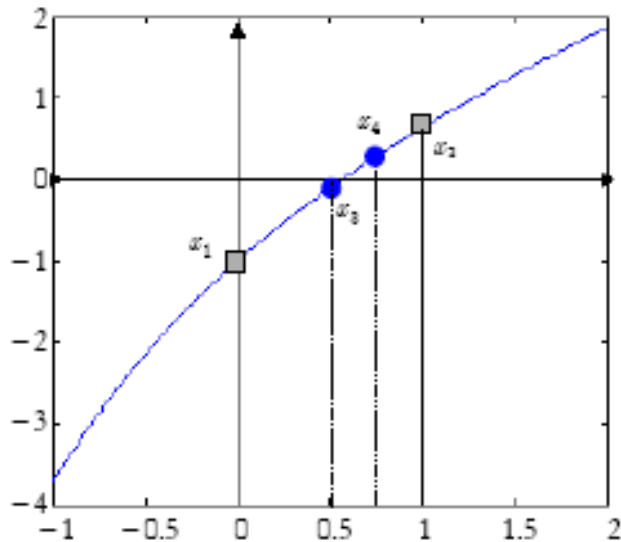
- 
- 예제) 구간 $(0,1)$ 에서 $f(x) = x - e^{-x}$ 의 실근을 이분법을 이용하여 구하라.

풀이. 초기값을 $x_1 = 0$ 과 $x_2 = 1$ 로 가정시,
 $f(0)f(1) < 0$ 이므로 근은 0과 1 사이에 존재한다.
따라서

$$x_3 = \frac{x_1 + x_2}{2} = \frac{0 + 1}{2} = 0.5$$

또한 $f(0)f(0.5) > 0$, $f(0.5)f(1) < 0$ 이므로 근이 0.5와 1 사이에 존재함을 알 수 있다.

반복의 종료를 위한 임계치를 0.0001로 했을 때 반복 결과 10회의 반복으로 실제값인 0.5671에 가까워짐을 알 수 있음



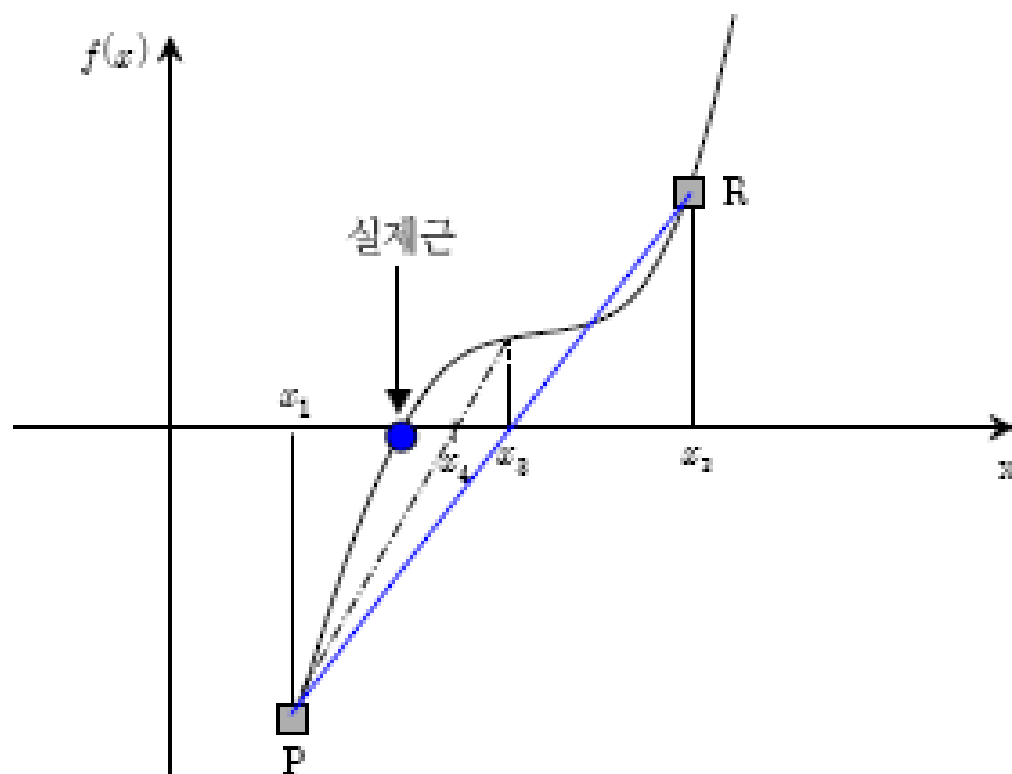
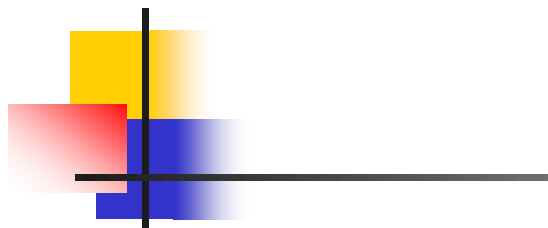
i(반복 횟수)	x_i	상대 오차(%)
0	0.0000	
1	0.5000	100
2	0.7500	33.3333
3	0.6250	20
4	0.5625	11.1111
5	0.5938	5.2632
6	0.5781	2.7027
7	0.5703	1.3699
8	0.5664	0.6897
9	0.5685	0.3436
10	0.5674	0.1721

7.4 가위치법 (false position method)

- 이분법의 수렴 속도를 개선한 것
- 이분법과 같이 구간을 절반으로 나누지 않고 $f(x_1)$ 과 $f(x_2)$ 을 직선으로 연결시켜 이 직선과 x 축이 만나는 교점 x_3 을 새로운 근으로 추정

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

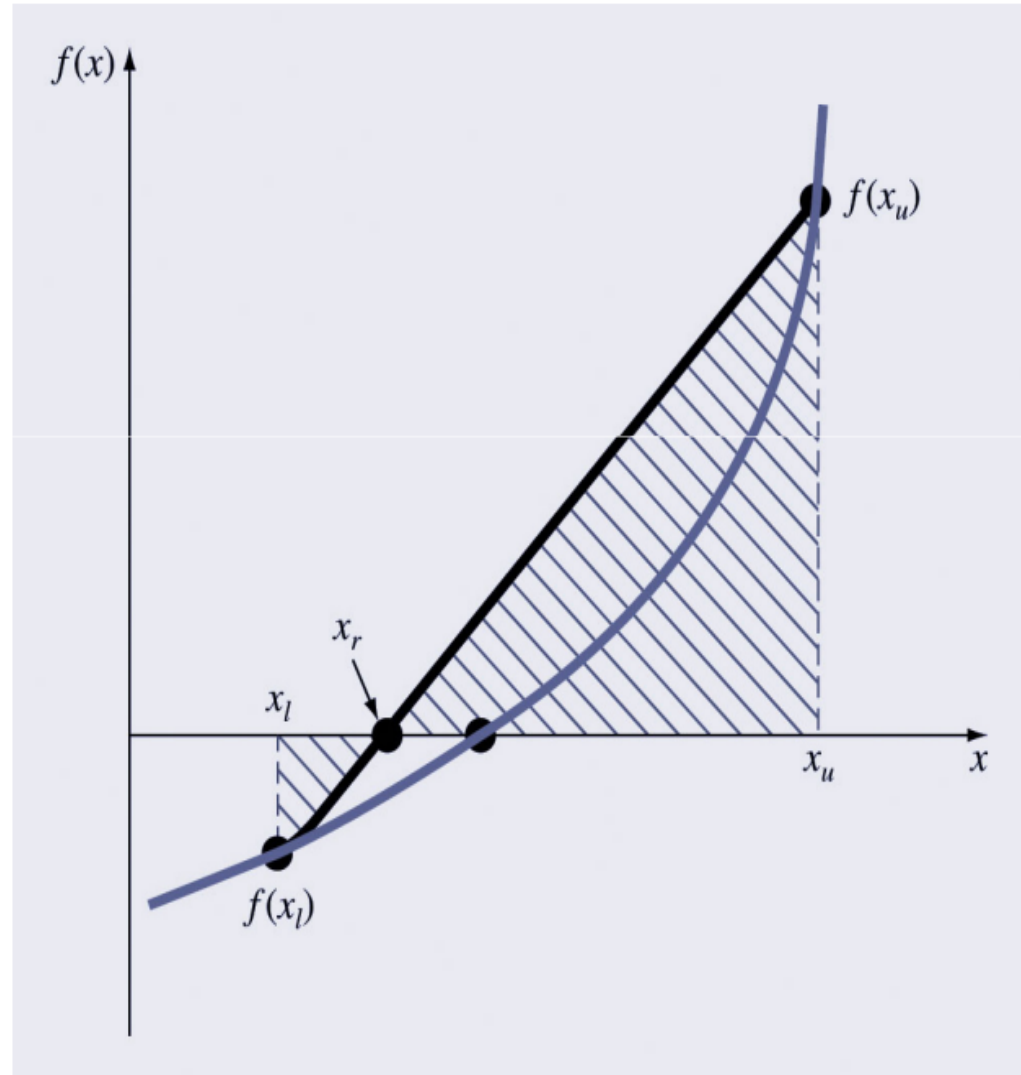
- if $f(x_1)f(x_3) < 0$ 이면 근은 x_1 와 x_3 사이에 존재
else 근은 x_2 와 x_3 사이에 존재
- 다시 두 점을 지나는 직선을 그어 x 축과 만나는 점을 다음 반복의 시작점으로 둬

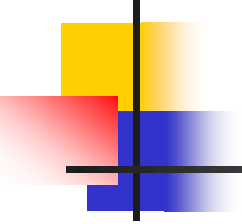


가위치법의 원리

- 이분법과 매우 유사하다.
- $f(x_l)$ 과 $f(x_u)$ 를 연결하는 직선과 x 축의 교점을 찾아 개선된 추정값으로 이용.
- 가위치법 공식

$$x_r = x_u - \frac{f(x_u)(x_l - x_u)}{f(x_l) - f(x_u)}$$

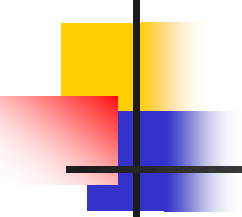


- 
- 예제) 구간 $(0, 1)$ 에서 $f(x) = x - e^{-x}$ 의 실근을 가위치법을 이용하여 구하라.

풀이. 초기값을 $x_1 = 0$ 과 $x_2 = 1$ 로 가정시,
 $f(0)f(1) < 0$ 이므로 근은 0과 1 사이에 존재한다.
따라서

$$x_3 = 1 - 0.6321 \left(\frac{1}{1.6321} \right) = 0.6127$$

또한 $f(0)f(0.6127) > 0$, $f(0.6127)f(1) < 0$ 이므로
근이 0.6127과 1 사이에 존재함을 알 수 있다.



0.6127과 1을 초기값으로 가정하고 임계치를 0.0001로
했을 때 4회 반복 결과

i(반복 횟수)	x_i	상대 오차(%)
0	0.0000	
1	0.6127	100
2	0.5722	7.0814
3	0.5677	0.7888
4	0.5672	0.0877