

Theory

○○○○○○○○

1. Interval Bisection

○○○○○

2. Fixed-Point Iteration

○○○○○○○

3. Newton's Method

○○○○○○

4. Secant Method

○○○○○○○

Finding a Root for Nonlinear Equation

Numerical Methods

Sim, Min Kyu, Ph.D., mksim@seoultech.ac.kr



1 Theory

2 1. Interval Bisection

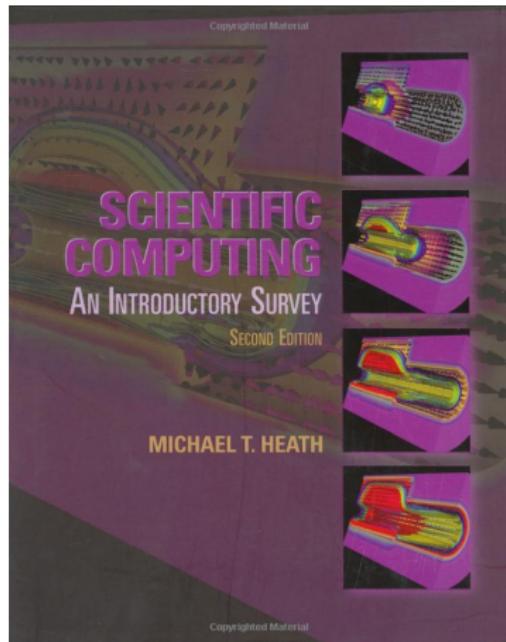
3 2. Fixed-Point Iteration

4 3. Newton's Method

5 4. Secant Method

About

- This note discusses how to find a root of nonlinear equations using numerical methods.
- This note is based on Chapter 5. Nonlinear Equation of the following book.
- Heath, M. T. (2018). Scientific computing: an introductory survey, revised second edition. Society for Industrial and Applied Mathematics.



Theory

●○○○○○○○

1. Interval Bisection

○○○○○

2. Fixed-Point Iteration

○○○○○○○

3. Newton's Method

○○○○○○

4. Secant Method

○○○○○○○

Theory

Definition: Root

- We are interested in finding values of x that solves an equation $f(x) = 0$ where $f(\cdot)$ is *nonlinear* function.
- Such a solution value x is called **solution** or **root** of the equation.

Number of solutions

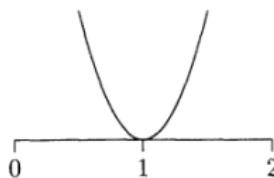
- A number of solution for a nonlinear equation may vary.
- Example
 - 1. $x^2 - 4\sin(x) = 0$ has a unique solution. ($x = 1.93375$)
 - 2. $e^x + 1 = 0$ has no solution.
 - 3. $x^2 - 4\sin(x) = 0$ has two solutions.
 - 4. $x^3 + 6x^2 + 11x - 6 = 0$ has three solutions.
 - 5. $\sin(x) = 0$ has infinitely many solutions.

Simple root and multiple root

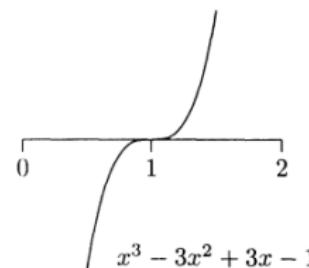
- If $f(x^*) = 0$ and $f'(x^*) \neq 0$, then x^* is said to be a **simple root**.
- A solution that satisfies both equation $f(x^*) = 0$ and $f'(x^*) = 0$ called a **multiple root**.
- If $f(x^*) = f'(x^*) = 0$ but $f''(x^*) \neq 0$, then x^* is said to be a *multiplicity 2*.
- If $f(x^*) = f'(x^*) = f''(x^*) = 0$ but $f'''(x^*) \neq 0$, then x^* is said to be a *multiplicity 3*.

Multiple roots

- The quadratic equation $x^2 - 2x + 1 = 0$ has a root of multiplicity two, $x = 1$.
- The cubic equation $x^3 - 3x^2 + 3x - 1 = 0$ has a root of multiplicity three, $x = 1$.



$$x^2 - 2x + 1$$



$$x^3 - 3x^2 + 3x - 1$$

Figure 5.2: Two nonlinear functions, each having a multiple root.

- There is no *sign change* around the root of left figure. This limits applicable numerical methods.

Theorem: Intermediate Value Theorem

- If f is continuous on a closed interval $[a, b]$, and c lies between $f(a)$ and $f(b)$, then there is a value $x^* \in [a, b]$ s.t. $f(x^*) = c$.
- If $f(a)$ and $f(b)$ differ in sign, then by taking $c = 0$ in the theorem we can conclude that there must be a *root* within the interval $[a, b]$.
- Such an interval $[a, b]$ for which the sign of f differs contains a solution.

Definition: Contractive mapping

- A function $g : \mathbb{R} \rightarrow \mathbb{R}$ is **contractive** on a set $S \subseteq \mathbb{R}$, if there is a constant γ , with $0 < \gamma < 1$, such that $\|g(x) - g(z)\| \leq \gamma \|x - z\|$ for all $x, z \in S$.

Definition: Fixed point

- A **fixed point** of g is any value x such that $g(x) = x$.

Theorem: Contractive mapping theorem

- If g is contractive on a closed set $S \subseteq \mathbb{R}$ and $g(S) \subseteq S$, then g has a unique fixed point in S .
- Thus, if f has the form $f(x) = x - g(x)$, where g is contractive on a closed set $S \subseteq \mathbb{R}$, with $g(S) \subseteq S$, then $f(x) = 0$ has a unique solution in S , namely the fixed point of g .

A sketch of fixed-point iteration

- In order to find a root for $f(x) = 0$, fixed-point algorithm goes as follows.
 - i) Express the equation in a form of $f(x) = x - g(x)$
 - ii) (where $g(x)$ is contractive)
 - iii) Then, find the fixed point of g , i.e. $g(x) = x$.
 - iv) The fixed point of g solves $f(x) = 0$.

Goal

- Given a continuous, nonlinear, and univariate function $f : \mathbb{R} \rightarrow \mathbb{R}$, we seek a point $x^* \in \mathbb{R}$ s.t. $f(x^*) = 0$.
- This note presents the following four solution methods.
 1. Interval Bisection
 2. Fixed-Point Iteration
 3. Newton's Method
 4. Secant Method

Theory

○○○○○○○

1. Interval Bisection

●○○○○

2. Fixed-Point Iteration

○○○○○○○

3. Newton's Method

○○○○○○

4. Secant Method

○○○○○○○

1. Interval Bisection

Motivation

- We seek a *very short interval* $[a, b]$ in which f has a change of sign at both ends.
- Since the function is continuous, the intermediate value theorem guarantees the root is contained in the interval $[a, b]$.

Interval bisection method

- i) begins with an initial interval that contains a root and
- ii) successively halves the interval
- iii) until the interval is short enough (i.e. $< tol$)

```
## 1. Interval Bisection
while (b-a > tol) do
  m=a+(b-a)/2 # m is midpoint of [a,b]
  if sgn(f(a))=sgn(f(m)) then
    a=m # so that [m,b] becomes new interval
  else
    b=m # so that [a,m] becomes new interval
  end
end
```

Example. $f(x) = x^2 - 4\sin(x) = 0$

- With $a=1$, $b=3$, and $\text{tol}=0.01$

```
f <- function (x) { return(x^2 - 4*sin(x)) }; a = 1; b = 3; tol = 0.01
print(paste0("Initial interval: [", a, ",", b, "]"))

"Initial interval: [1,3]"
while (b-a > tol) {
  m <- a+(b-a)/2
  if (f(a)*f(m)>0) {
    a <- m
  } else {
    b <- m
  }
  print(paste0("Current interval: [", a, ",", b, "]"))
}

"Current interval: [1,2]"
"Current interval: [1.5,2]"
"Current interval: [1.75,2]"
"Current interval: [1.875,2]"
"Current interval: [1.875,1.9375]"
"Current interval: [1.90625,1.9375]"
"Current interval: [1.921875,1.9375]"
"Current interval: [1.9296875,1.9375]"
```

Discussion

- The bisection method makes no use of function value except for the signs.
- This makes convergence rate low.
- With an initial interval $[a, b]$, length of interval after k -th iteration is $(b - a)/2^k$.

Exercise. Use the above method to find that $x = 0.567$ approximately solves a nonlinear equation $f(x) = e^{-x} - x = 0$.

7Q

Theory

○○○○○○○○

1. Interval Bisection

○○○○○

2. Fixed-Point Iteration

●○○○○○○

3. Newton's Method

○○○○○○

4. Secant Method

○○○○○○○○

2. Fixed-Point Iteration

Motivation

- Remind that the interval bisection method does not make use of function value except for signs. This results in the lower convergence rate.
- The fixed-point iteration does make use of function values.

Development

- Remind that, for a function $g : \mathbb{R} \rightarrow \mathbb{R}$, a value x such that $x = g(x)$ is called a *fixed point* of the function g .
- For any given equation $f(x) = 0$, we can convert the problem into a form of $x = g(x)$, where the function g is *contractive* at the solution x^* .

Preparation of $x = g(x)$

- One needs to convert the original problem $f(x) = 0$ into the form of $x = g(x)$. This conversion process is not difficult.
- For example, by letting $g(x) := f(x) + x$, the original problem is converted to a fixed point problem of $x = g(x)$.
- The conversion is not unique, as will be discussed.

Fixed-Point Iteration method

- i) Rearrange the equation $f(x) = 0$ in the form of $x = g(x)$.
- ii) Choose the following.
 - x_0 : initial guess of solution
 - tol : tolerable error
 - N : maximum iterations
- iii) Repeat the iteration scheme $x_{k+1} = g(x_k)$ until $|f(x)| \leq tol$ or $\text{iter} > N$

2. Fixed Point Iteration

```

while (f(x_old) > tol) or (iter <= N) do
  x_new <- g(x_old) # repeat the iteration scheme
  x_old <- x_new
  iter <- iter + 1
end

```

Example. $f(x) = x^2 - x - 2 = 0$

- The equation $x^2 - x - 2 = 0$ is converted to $x^2 = x + 2$, which is converted to $x = 1 + 2/x$.
- Thus, we let $g(x) = 1 + 2/x$
- With the setting of $x_0=1$, $tol=0.01$, and $N=5$

```
f <- function (x) { return (x^2-x-2) }
g <- function (x) { return (1+2/x) }
x_old <- 1;
tol <- 0.01
N <- 5;
iter <- 0;
print(paste0("x_old:", x_old,
            " f(x_old):", f(x_old),
            " g(x_old):", g(x_old)))

```

"x_old:1 f(x_old):-2 g(x_old):3"

```
while ((abs(f(x_old)) > tol) || (iter <= N)) {
  x_new <- g(x_old) # repeat the iteration scheme
  x_old <- x_new
  iter <- iter + 1
  print(paste0("x_old:", round(x_old, 3),
              " f(x_old):", round(f(x_old), 3),
              " g(x_old):", round(g(x_old), 3)))
}
```

```
"x_old:3 f(x_old):4 g(x_old):1.667"
"x_old:1.667 f(x_old):-0.889 g(x_old):2.2"
"x_old:2.2 f(x_old):0.64 g(x_old):1.909"
"x_old:1.909 f(x_old):-0.264 g(x_old):2.048"
"x_old:2.048 f(x_old):0.145 g(x_old):1.977"
"x_old:1.977 f(x_old):-0.069 g(x_old):2.012"
"x_old:2.012 f(x_old):0.035 g(x_old):1.994"
"x_old:1.994 f(x_old):-0.018 g(x_old):2.003"
"x_old:2.003 f(x_old):0.009 g(x_old):1.999"
```

Convergence and divergence

- If $x^* = g(x^*)$ and $|g'(x^*)| < 1$, then the iterative scheme is *locally convergent*, i.e., there is an interval containing x^* s.t. fixed-point iteration with g **converges** if started at a point within that interval.
- If $|g'(x^*)| > 1$, then fixed-point iteration with g **diverges**.

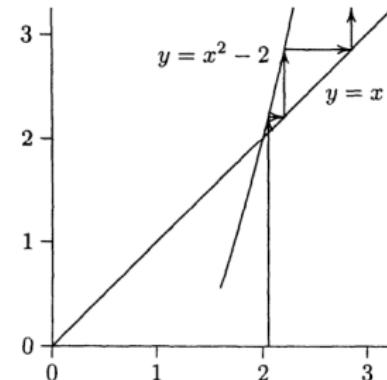
Four different approaches for $f(x) = x^2 - x - 2 = 0$

$$1. \ x^2 - x - 2 = 0 \Rightarrow x = x^2 - 2$$

$$\Rightarrow g_1(x) = x^2 - 2$$

- $g'_1(x) = 2x, g'_1(2) = 4$

- It diverges because $|g'_1(2)| > 1$.

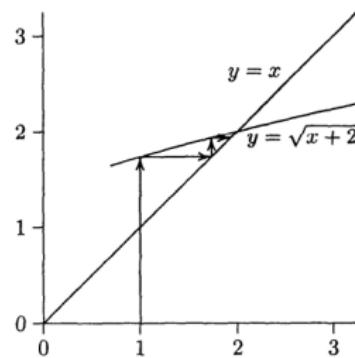


$$2. \ x^2 = x + 2 \Rightarrow x = \sqrt{x+2} \Rightarrow$$

$$g_2(x) = \sqrt{x+2}$$

- $g'_2(x) = \frac{1}{(2\sqrt{x+2})}, g'_2(2) = \frac{1}{4}$

- It converges linearly.



$$3. \quad x^2 = x + 2 \implies x = 1 + \frac{2}{x} \implies$$

$$g_3(x) = 1 + \frac{2}{x}$$

- $g'_3(x) = \frac{-2}{x^2}, g'_3(2) = -\frac{1}{2}$

- It converges linearly.

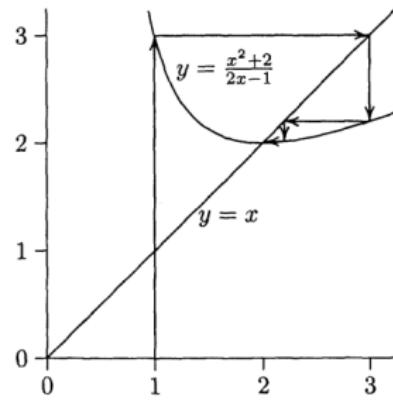
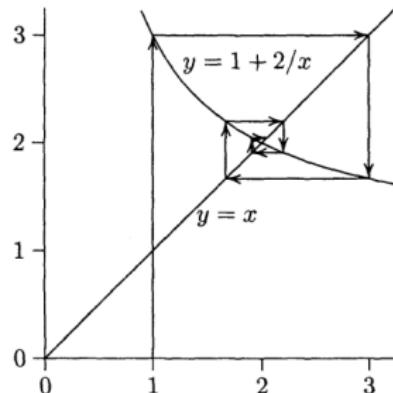
$$4. \quad 2x^2 - x^2 - x - 2 = 0 \implies$$

$$2x^2 - x = x^2 + 2 \implies x = \frac{x^2+2}{2x-1}$$

$$\implies g_4(x) = \frac{x^2+2}{2x-1}$$

- $g'_4(x) = \frac{2x^2-2x-4}{(2x-1)^2}, g'_4(2) = 0$

- It converges quadratically.



Theory

○○○○○○○○

1. Interval Bisection

○○○○○

2. Fixed-Point Iteration

○○○○○○○

3. Newton's Method

●○○○○○○

4. Secant Method

○○○○○○○○

3. Newton's Method

Motivation

- Taylor's 1st order expansion goes as follows.

$$f(x + h) \approx f(x) + f'(x) \cdot h$$

- We can replace the *nonlinear function* f with *linear function* using its derivative.

Development

- We can view Newton's method as a systematic way of transforming a nonlinear equation $f(x) = 0$ into a fixed-point problem $x = g(x)$, where

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

- This method approximates the function f near x_k by the tangent line at $f(x_k)$.
- Then, the root of this tangent line becomes the next approximate solution.
- Repeat this process.

Newton's Method

- i) Rearrange the equation $f(x) = 0$ in the form of $x = g(x)$, where

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

- ii) Choose the followings.

- x_{old} : initial guess of solution
- tol : tolerable error
- N : maximum iterations

- iii) Repeat the iteration scheme $x_{k+1} = g(x_k)$ until $|f(x)| \leq \text{tol}$ or $\text{iter} > N$

```
## 3. Newton's Method
x_old <- initial guess
while (|f(x)| > tol) or (iter <= N) do
  x_new <- x_old - f(x_old)/f'(x_old)
  iter <- iter+1
  x_old <- x_new
end
```

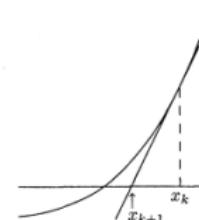


Figure 5.6: Newton's method for solving nonlinear equation.

Example. $f(x) = x^2 - 4\sin(x) = 0$

- With $x_{\text{old}}=3$, $\text{tol}=0.001$

```
f <- function (x) { return(x^2 - 4*sin(x)) }
df <- function (x) { return(2*x - 4*cos(x)) }
x_old <- 3; tol <- 0.001; N <- 3; iter <- 0;
print(paste0("x:", x_old, " f(x):", f(x_old)))

"x:3 f(x):8.43551996776053"
while((abs(f(x_old))>tol) & (iter<=N)){
  x_new <- x_old - f(x_old)/df(x_old)
  x_old <- x_new
  iter <- iter + 1
  print(paste0("x:", x_new, " f(x):", f(x_new)))
}

"x:2.15305769201339 f(x):1.29477250528657"
"x:1.9540386420058 f(x):0.108438553394625"
"x:1.93397153275207 f(x):0.00115163152386399"
"x:1.93375378855763 f(x):0.000000136054946420217"
print(paste0("The root is ", x_new))

The root is 1.93375378855763"
```

Discussion

- The Newton's method has its drawback that both the function and its derivative must be evaluated at each iteration.
- Newton's method is fast, but requires its derivative analytically available.

Theory

○○○○○○○

1. Interval Bisection

○○○○○

2. Fixed-Point Iteration

○○○○○○○

3. Newton's Method

○○○○○●

4. Secant Method

○○○○○○○

Exercise. Use the above method to find that $x = 0.567$ approximately solves a nonlinear equation $f(x) = e^{-x} - x = 0$.

7Q

Theory

○○○○○○○○

1. Interval Bisection

○○○○○

2. Fixed-Point Iteration

○○○○○○○○

3. Newton's Method

○○○○○○○

4. Secant Method

●○○○○○○○

4. Secant Method

Motivation

- A derivative of a function may be inconvenient or expensive to evaluate.
- So, secant method uses a better idea that is to use the finite difference approximation instead.

$$f'(x_k) \approx \frac{f(x_k) - f(x_{k-1})}{x_k - x_{k-1}}$$

- The secant method can be interpreted as approximating the function f by the secant line through the previous two iterations.
- Then take the root of the resulting linear function to be the next approximate solution.

Secant Method

i) Choose the followings.

- x_{old} : 0-th initial guess of solution
- x_{new} : 1-st initial guess of solution
- tol : tolerable error
- N : maximum iterations

ii) Repeat the iteration scheme

$$x_{\text{next}} = x_{\text{new}} - f(x_{\text{new}}) \cdot \frac{x_{\text{new}} - x_{\text{old}}}{f(x_{\text{new}}) - f(x_{\text{old}})}$$

and update x_{old} and x_{new} until $|x_{\text{new}} - x_{\text{old}}| \leq \text{tol}$ or $\text{iter} > N$

4. Secant Method

```

x_old <- 0th initial guess
x_new <- 1st initial guess
while (|x_new-x_old| > tol) or (iter <= N) do
  x_next = x_new-f(x_new)*(x_new-x_old)/(f(x_new)-f(x_old))
  x_old = x_new
  x_new = x_next
end

```

Example. $f(x) = x^2 - 4\sin(x) = 0$

- With $x_{\text{old}}=1$, $x_{\text{new}}=3$

```
f <- function (x) { return(x^2 - 4*sin(x)) }
x_old <- 1; x_new <- 3; tol <- 0.1; N <- 5; iter <- 0;
print(paste0(iter, "-th iter: ", "x_old:", x_old, " x_new:", x_new, " f(x_old):", f(x_old)))

"0-th iter: x_old:1 x_new:3 f(x_old):-2.36588393923159"

while ((abs(x_new-x_old) > tol) & (iter <= N)) {
  x_next = x_new-f(x_new)*(x_new-x_old)/(f(x_new)-f(x_old))
  x_old = x_new
  x_new = x_next
  iter <- iter+1
  print(paste0(iter, "-th iter: ", "x_old:", x_old, " x_new:", x_new, " f(x_old):", f(x_old)))
}

"1-th iter: x_old:3 x_new:1.43806971012353 f(x_old):8.43551996776053"
"2-th iter: x_old:1.43806971012353 x_new:1.72480462104936 f(x_old):-1.89677449157582"
"3-th iter: x_old:1.72480462104936 x_new:2.02983325288416 f(x_old):-0.977705597349626"
"4-th iter: x_old:2.02983325288416 x_new:1.92204417896096 f(x_old):0.534304487516024"
"5-th iter: x_old:1.92204417896096 x_new:1.93317401864344 f(x_old):-0.0615225574089555"

print(paste0("The root is ", x_new))

"The root is 1.93317401864344"
```

Discussion

- Compared with Newton's method, the secant method has
 - the advantage of requiring only one new function evaluation per iteration.
 - the disadvantages of requiring two starting guesses and converging somewhat more slowly.
- By using secant method, there is no need for the process of rearranging the function $f(x)$ to $g(x)$.

Theory

○○○○○○○○

1. Interval Bisection

○○○○○

2. Fixed-Point Iteration

○○○○○○○

3. Newton's Method

○○○○○○

4. Secant Method

○○○○○●○

Exercise. Use the above method to find that $x = 0.567$ approximately solves a nonlinear equation $f(x) = e^{-x} - x = 0$.

7Q

Acknowledgement

- The lecture note is made possible thanks to the hard work by Yujeong Hwang (ITM 20'). She served as the school's ambassador "Aumi" and she was a second-year ITM representative. In the Applied Probability Lab, she conducted research about Reinforcement Learning for DC-DC converter.

