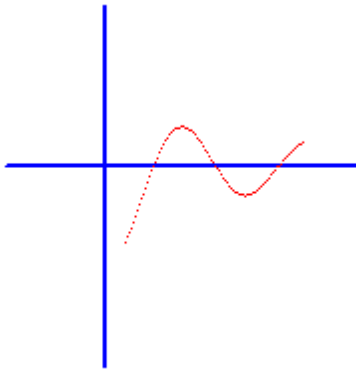


Bessj0 function은  $(1 \leq x \leq 10)$ 에서 다음과 같이 생겼다



그래프를 보면 근이 세 개 있음을 알 수 있는데 여러 method를 통해서 그 값을 추정할 수 있다.

#### 1. Bisection

```
which method?
=====
1. bisection
2. linear Interpolation
3. newton-raphson
4. newton With Bracketing
5. secant
6. muller
7. otherfunc
0. exit
=====
1

Roots of bessj0:
      x      f(x)
root 1      2.404826 -0.000000
root 2      5.520073 -0.000002
root 3      8.653730 -0.000001
```

#### 2. Linear interpolation

```
which method?
=====
1. bisection
2. linear Interpolation
3. newton-raphson
4. newton With Bracketing
5. secant
6. muller
7. otherfunc
0. exit
=====
2

Roots of bessj0:
      x      f(x)
root 1      2.404826 -0.000000
root 2      5.520078  0.000000
root 3      8.653728  0.000000
```

### 3. newton-rapson

```
which method?
=====
1. bisection
2. linear Interpolation
3. newton-rapson
4. newton With Bracketing
5. secant
6. muller
7. otherfunc
0. exit
=====
3

Roots of bessj0:
      x      f(x)
root 1  2.404825  0.000000
root 2  5.520078  0.000000
root 3  8.653728  0.000000
```

### 4. newton with bracketing

```
which method?
=====
1. bisection
2. linear Interpolation
3. newton-rapson
4. newton With Bracketing
5. secant
6. muller
7. otherfunc
0. exit
=====
4

Roots of bessj0:
      x      f(x)
root 1  2.404825  0.000000
root 2  5.520078  0.000000
root 3  8.653728  0.000000
```

### 5. secant

```
which method?
=====
1. bisection
2. linear Interpolation
3. newton-rapson
4. newton With Bracketing
5. secant
6. muller
7. otherfunc
0. exit
=====
5

Roots of bessj0:
      x      f(x)
root 1  2.404825  0.000000
root 2  5.520078  0.000000
root 3  8.653728  0.000000
```

## 6. muller

```

which method?
=====
1. bisection
2. linear Interpolation
3. newton-raphson
4. newton With Bracketing
5. secant
6. muller
7. otherfunc
0. exit
=====
6

Roots of bessj0:
      x      f(x)
root 1  33.775822 -0.000000
root 2  65.189964  0.000000
root 3  18.071064  0.000000

```

살펴보면 bisection에서 약간의 오차가 있고 나머지 방법들은 오차가 거의 없이 정확한 것을 확인할 수 있고, muller method 에서는 해당 범위에서 근이 아닌 다른 근을 찾았는데 해당 범위의 근을 찾는 값을 대입해야 하는데 그 값을 추정하는데 어려움이 있었다.

이론상 muller method 는 quadratic convergence이기 때문에 가장빠르고 다른 method들과 차이가 있지만 계산량이 너무 적어서 시간 차이를 측정하는데는 어려움이 있었다.

그 밖에 다른 function들은 다음과 같다.

```

which method?
=====
1. bisection
2. linear Interpolation
3. newton-raphson
4. newton With Bracketing
5. secant
6. muller
7. otherfunc
0. exit
=====
7

Roots of fx1
      x      f(x)
root 1  0.449261 -0.000000

Roots of fx2
      x      f(x)

Roots of fx3
      x      f(x)

Roots of fx4
      x      f(x)
root 1  -0.321751 -0.000000
root 2   0.000000  0.000000

```

$$F_{x1} = 10e^{-x}\sin(2\pi x) - 2$$

$$F_{x2} = x^2 - 2xe^{-x} + e^{-2x}$$

$$F_{x3} = \cos(x + \sqrt{2}) + x\left(\frac{x}{2} + \sqrt{2}\right)$$

$$F_{x4} = e^{-x}(3\cos 2x + 4\sin 2x)x$$

와 같다

$F_{x2}$ 와  $f_{x3}$ 은 근의 양쪽 범위에서 모두 양수이기 때문에 bracketing method로 범위를 추정할 수 없어 근을 못 구하는 듯 하였다.

한계점

- muller method 에서 범위에 해당하는 근을 구하기 위해 넣어야 할 초기값을 찾는 방법
- bisection method에서 약간의 오차 발생
- 구하는 과정을 그래프로 표현하는 방법에 대한 어려움