Assignment 2b

Yinqi Zhao

2/23/2020

Secant Method

1. Conducting the Secant method

Here is a function which will conduct the Secant method.

```
# secant method
secant <- function(fx, start0, start1, tol = 1e-3, maxitr = 1000){
  res <- c(start0, start1)
  distance <- -Inf
  x0 <- start0
  x1 <- start1
  i <- 1
  while(i <= maxitr & abs(distance) > tol){
    x2 <- x1 - fx(x1) * (x1 - x0) / (fx(x1) - fx(x0))
    res <- c(res, x2)
    distance <- abs(x2 - x1)
    x0 <- x1
    x1 <- x2
    print(paste0("iteration ", i, ": ", "X = ", x2, ", Y = ", fx(x2)))
    i <- i + 1
  }
  if(abs(distance) > tol){
    print("fail to converge, please try another start value")
  }
  print(paste0("converged! the root is ", x1))
  return(list(root = x2,
              res = res))
}
```

We test it on 2 functions, the first one is $f(x) = \cos x - x$, with start value $x_0 = 1$ and $x_1 = 2$.

```
f1 <- function(x){
  cos(x) - x
}

res.secant1 <- secant(fx = f1, start0 = 1, start1 = 2)
```

```
## [1] "iteration 1: X = 0.765034682391819, Y = -0.0436763442286054"
## [1] "iteration 2: X = 0.742299406864944, Y = -0.00538326126319721"
## [1] "iteration 3: X = 0.739103270158936, Y = -3.03543288344699e-05"
## [1] "iteration 4: X = 0.73908514606566, Y = -2.15067506026401e-08"
## [1] "converged! the root is 0.73908514606566"
```

The second function is $f(x) = \log x - \exp(x)$, with the same start value as above.

```
f2 <- function(x){
  log(x) - exp(-x)
}

res.secant2 <- secant(fx = f2, start0 = 1, start1 = 2)
```

```
## [1] "iteration 1: X = 1.39741048216961, Y = 0.0873845096214802"
## [1] "iteration 2: X = 1.28547612015065, Y = -0.0253897248274014"
## [1] "iteration 3: X = 1.31067675808254, Y = 0.000906097784013626"
## [1] "iteration 4: X = 1.3098083980193, Y = 9.10606693577121e-06"
## [1] "converged! the root is 1.3098083980193"
```

2. Comparing to the Newtons method.

A function to conduct the Newton's method.

```
newton <- function(fx, fx2, start, tol = 1e-3, maxitr = 1000){
  res <- start
  distance <- -Inf
  x1 <- start
  i <- 1
  while(i <= maxitr & abs(distance) > tol){
    x2 <- x1 - fx(x1) / fx2(x1)
    res <- c(res, x2)
    distance <- abs(x2 - x1)
    x1 <- x2
    print(paste0("iteration ", i, ": ", "X = ", x1, ", Y = ", fx(x1)))
    i <- i + 1
  }
  if(abs(distance) > tol){
    print("fail to converge, please try another start value")
  }
  print(paste0("converged! the root is ", x1))
  return(list(root = x2,
              res = res))
}

f1.d <- function(x){
  -sin(x) - 1
}

f2.d <- function(x){
  1 / x + exp(-x)
}
```

Choose 0 as the start point of Newton's method.

```
invisible(capture.output(res.newton1 <- newton(f1, f1.d, start = 1)))
invisible(capture.output(res.newton2 <- newton(f2, f2.d, start = 1)))

results <- data.frame(Secant.itr = c(length(res.secant1$res), length(res.secant2$res)),
                      Secant.root = c(res.secant1$root, res.secant2$root),
                      Newton.itr = c(length(res.newton1$res), length(res.newton2$res)),
                      Newton.root = c(res.newton1$root, res.newton2$root))
```

```
row.names(results) <- c("funtion1", "function2")
results
```

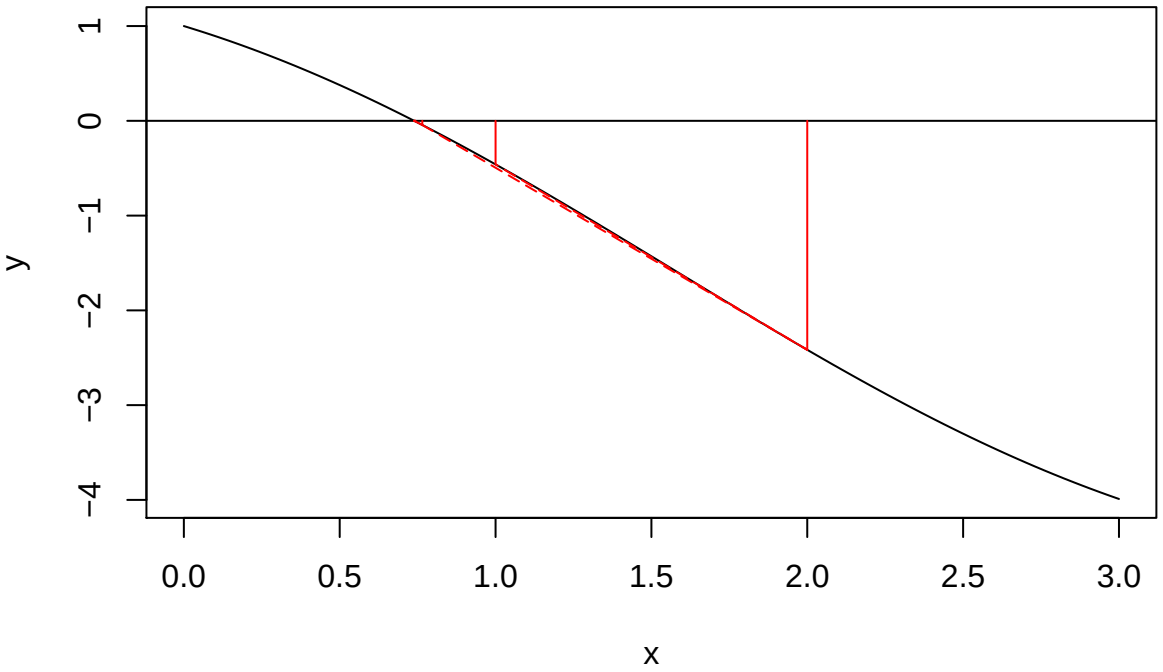
```
##           Secant.itr Secant.root Newton.itr Newton.root
## funtion1           6   0.7390851          4   0.7390851
## function2          6   1.3098084          4   1.3097994
```

Both root finding method produce the same results. It is not very straightforward to compare the convergence speed of the two method. Based on the initial value of 1 for Newton's method and 1, 2 for Secant method, Newton's method is a little bit faster than the Secant but the difference is not very big. Both method converge very fast

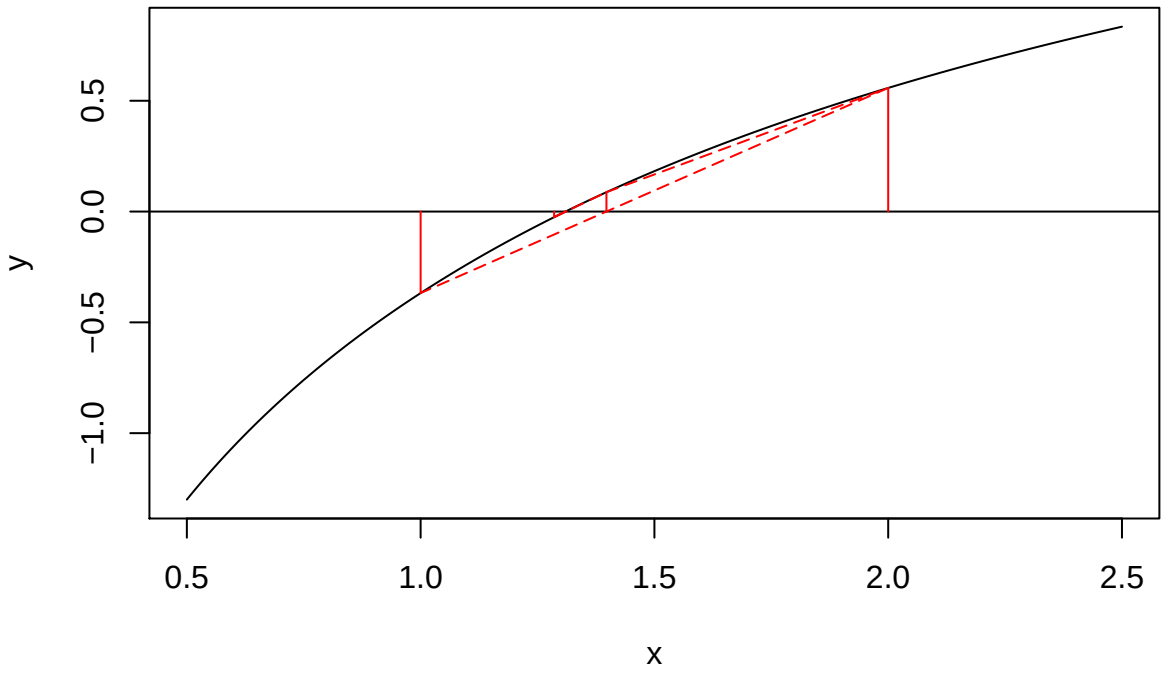
Plot the process of Secant method

```
plot.process <- function(fx, res, xrange = c(-1, 1)){
  grid <- seq(from = xrange[1], to = xrange[2], length.out = 1000)
  plot(grid, fx(grid), type = "l", xlab = "x", ylab = "y")
  abline(a = 0, b = 0)
  for(i in 1:length(res)){
    x <- res[i]
    segments(x0 = x, y0 = 0, x1 = x, y1 = fx(x), col = "red")
    if(i > 1){
      segments(x0 = x, y0 = fx(x), x1 = res[i - 1], y1 = fx(res[i - 1]), col = "red", lty = 5)
    }
  }
}

plot.process(f1, res.secant1$res, xrange = c(0, 3))
```




```
plot.process(f2, res.secant2$res, xrange = c(0.5, 2.5))
```


y
0.5
0.0
−0.5
−1.0
0.5
1.0
1.5
2.0
2.5
x