NewtonRaphson-Yina

Yina Liu

2/11/2020

```
# Define functions
# (All functions need to return f(x) and f'(x))
F1<-function(x){
  return(c(x^2,2*x)) # note that the function returns two numbers. The first is f(x); the second is the
}
#define a function F2(x)=sin(x)
F2<-function(x){
  return(c(sin(x),cos(x)))
}
#define F3(x)=(x-2)^3-6*x
F3<-function(x){
  return(c((x-2)^3-6*x,3*(x-2)^2-6))
}
#define F4(x)=cos(x)-x###
F4<-function(x){
  return(c(cos(x)-x,-sin(x)-1))
}

# Define Newton-Raphson function
library(shape)
NewtonRaphson<-function(func,StartingValue,Tolerance,MaxNumberOfIterations,DrawLines){
  #initialize a variable, Deviation (say), to record |f(x)| so that you know how far away you are from 
  #(So initialize it to some arbitrary large number)
  Deviation <- 1000
  #Set up a counter, i, to record how many iterations you have performed. Set it equal to 0
  i <- 0
  # Initialize the values of x and f(x)
  X  <- StartingValue

  #Set up a while loop until we hit the required target accuracy or the max. number of steps
  Z <- c()
  while ((i<MaxNumberOfIterations)&&(Deviation>Tolerance))
  {
    # Record the value of f(x) and f'(x), for the current x value.
    Xprime <- func(X)
    Z[1] <- Xprime[1]
    Z[2] <- Xprime[2]
    X_1 <- X - Z[1]/Z[2] #To draw line segment for Xn+1
    # I put them in a variable Z. Z[1]<-f(x); Z[2]<-f'(x)
    # To be safe, check that the function and it's derivative are defined at X (either could be NaN if 
    if ((Z[1]=="NaN")||(Z[2]=="NaN")){
      cat("\nFunction or derivative not defined error.\n")
```

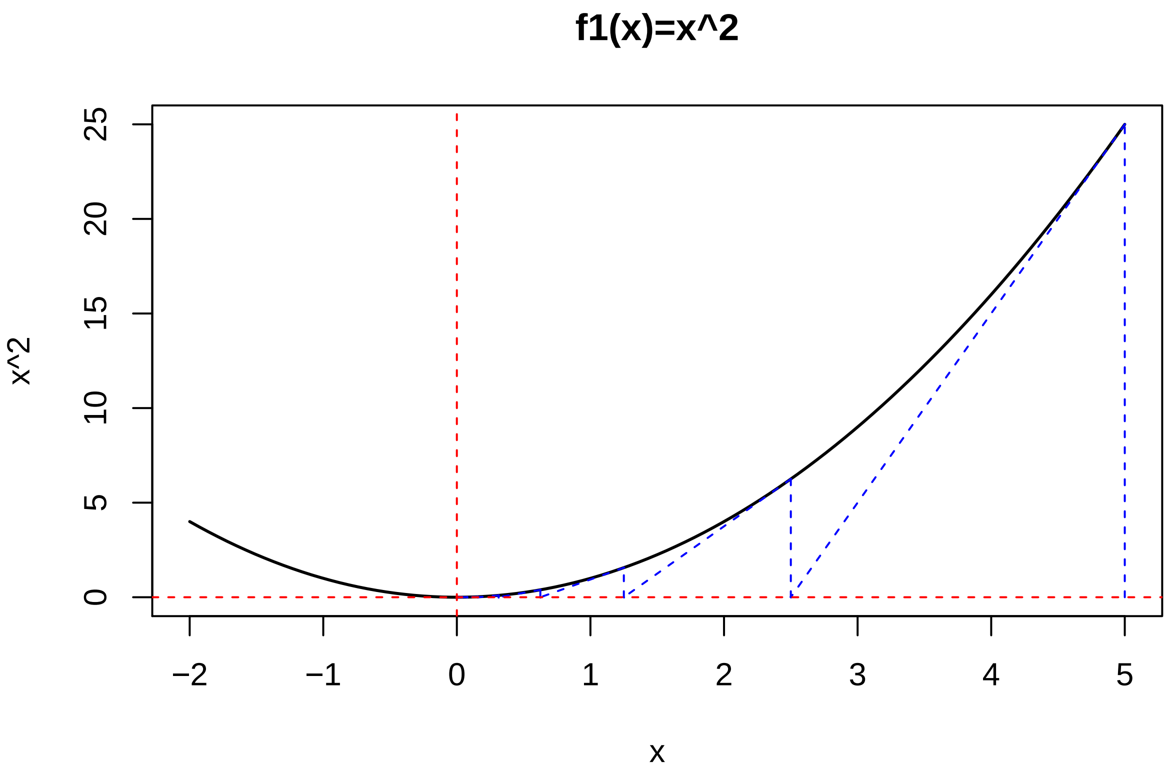
```
      break
    }
    if (DrawLines){
      Arrows(X,0,X,Z[1],col="blue",lty=2,arr.length=0.01, arr.type = "T")
      Arrows(X,Z[1],X_1,0,col="blue",lty=2,arr.length=0.01, arr.type = "T")
      }

    #Find the next X-value using Newton-Raphson's formula. Let's call that value X
    X <- X - Z[1]/Z[2]
    Y <- func(X)[1]
    # calculate Deviation<- |f(x)-0|
    Deviation <- abs(Z[1]-0)
    # increase the value of your iteration counter
    i <- i+1

    # if you like, have the program write out how it is getting on
    cat(paste("\nIteration ",i,":   X=",X,"  Y=",Y))

    # If you are feeling fancy, add some line segments to the screen to show where it just went
    # See the 'fixed points' code for a reminder of how to do that.
    # output the result
    if (Deviation<Tolerance){
      cat(paste("\nFound the root point: ",X, " after ", i, "iterations"))
    }else{
      cat(paste("\nConvergence failure. Deviation: ",Deviation, "after ", i,    "iterations"))}
  }

  # have the function return the answer
  return(X)
}

# Results and plots of Newton-Raphson

# Root of x^2
curve(x^2, xlim=c(-2,5), lwd=1.5, main="f1(x)=x^2")
NewtonRaphson(F1,5,1e-3,40,1)
```

```
##
## Iteration  1 :   X= 2.5   Y= 6.25
## Convergence failure. Deviation:  25 after  1 iterations
## Iteration  2 :   X= 1.25   Y= 1.5625
## Convergence failure. Deviation:  6.25 after  2 iterations
## Iteration  3 :   X= 0.625   Y= 0.390625
## Convergence failure. Deviation:  1.5625 after  3 iterations
## Iteration  4 :   X= 0.3125   Y= 0.09765625
## Convergence failure. Deviation:  0.390625 after  4 iterations
## Iteration  5 :   X= 0.15625   Y= 0.0244140625
## Convergence failure. Deviation:  0.09765625 after  5 iterations
## Iteration  6 :   X= 0.078125   Y= 0.006103515625
## Convergence failure. Deviation:  0.0244140625 after  6 iterations
## Iteration  7 :   X= 0.0390625   Y= 0.00152587890625
## Convergence failure. Deviation:  0.006103515625 after  7 iterations
## Iteration  8 :   X= 0.01953125   Y= 0.0003814697265625
## Convergence failure. Deviation:  0.00152587890625 after  8 iterations
```

```
## Iteration  9 :   X= 0.009765625   Y= 9.5367431640625e-05
## Found the root point:  0.009765625  after  9 iterations
```

```
## [1] 0.009765625
```

```r
abline(h=0,col='red',lwd=1, lty=2)
abline(v=0,col='red',lwd=1, lty=2)
```

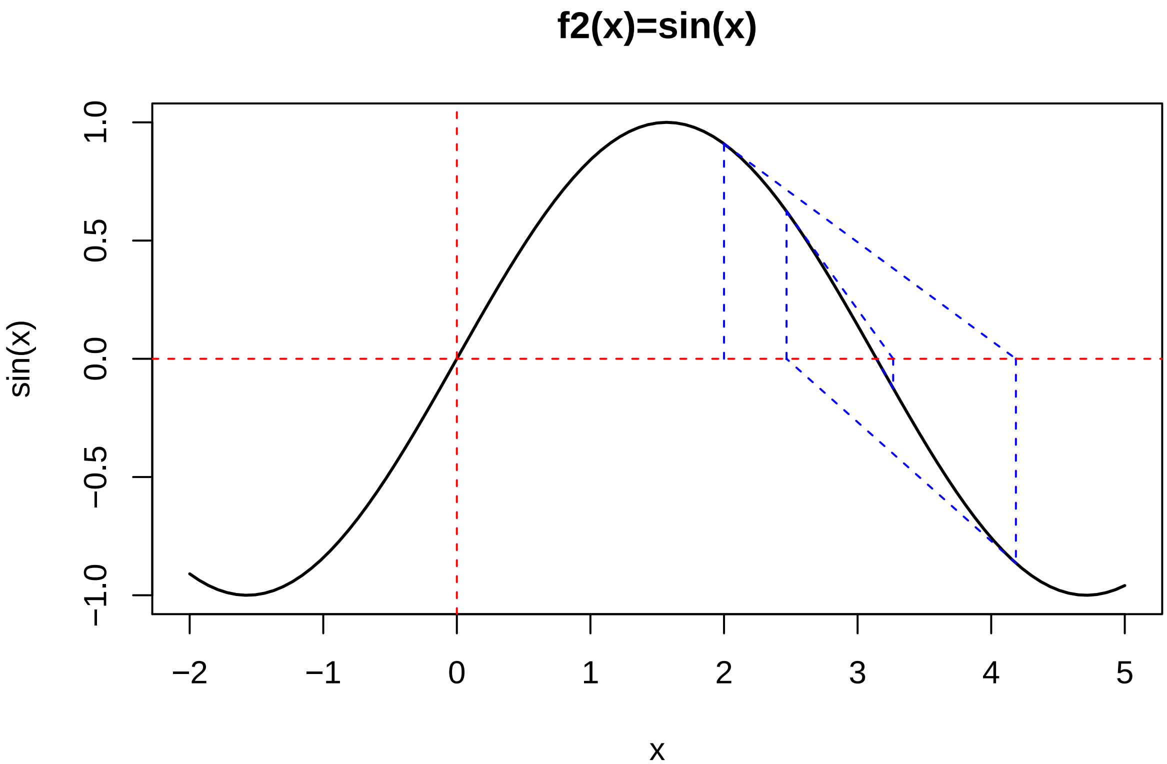



```r
# Root of sin(x)
curve(sin(x), xlim=c(-2,5), lwd=1.5, main="f2(x)=sin(x)")
NewtonRaphson(F2,2,1e-3,40,1)
```

```
##
## Iteration  1 :   X= 4.18503986326152   Y= -0.864144147074565
## Convergence failure. Deviation:  0.909297426825682 after  1 iterations
## Iteration  2 :   X= 2.46789367451467   Y= 0.623881072066777
## Convergence failure. Deviation:  0.864144147074565 after  2 iterations
## Iteration  3 :   X= 3.26618627756911   Y= -0.124271517762097
## Convergence failure. Deviation:  0.623881072066777 after  3 iterations
## Iteration  4 :   X= 3.14094391231764   Y= 0.000648741226652542
## Convergence failure. Deviation:  0.124271517762097 after  4 iterations
## Iteration  5 :   X= 3.1415926536808   Y= -9.10110761127821e-11
## Found the root point:  3.1415926536808  after  5 iterations
```

```
## [1] 3.141593
```

```r
abline(h=0,col='red',lwd=1, lty=2)
abline(v=0,col='red',lwd=1, lty=2)
```

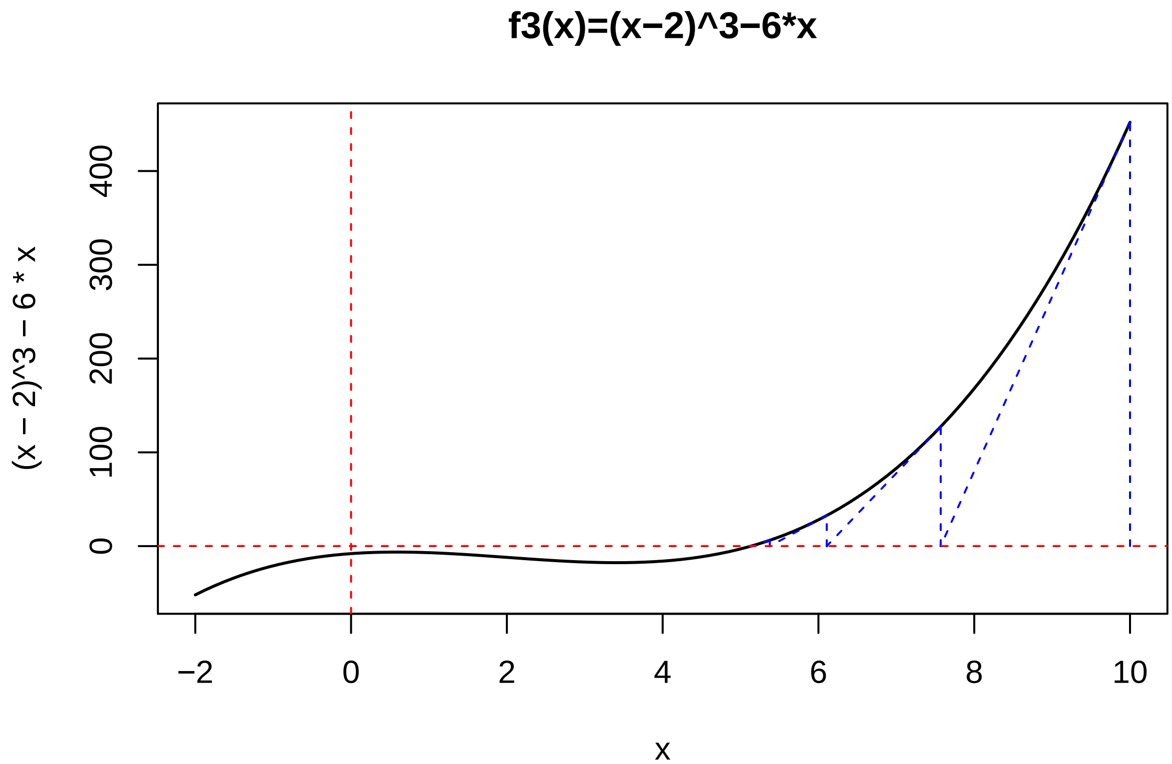



```r
# Root of (x-2)^3-6*x)
curve((x-2)^3-6*x, xlim=c(-2,10), lwd=1.5, main="f3(x)=(x-2)^3-6*x")
NewtonRaphson(F3,10,1e-3,40,1)
```

```
##
## Iteration  1 :   X= 7.56989247311828   Y= 127.379330322233
## Convergence failure. Deviation:  452 after  1 iterations
## Iteration  2 :   X= 6.10695791925136   Y= 32.6307361646652
## Convergence failure. Deviation:  127.379330322233 after  2 iterations
## Iteration  3 :   X= 5.3753485533609   Y= 6.20318000683711
## Convergence failure. Deviation:  32.6307361646652 after  3 iterations
## Iteration  4 :   X= 5.15521318197001   Y= 0.48003626931828
## Convergence failure. Deviation:  6.20318000683711 after  4 iterations
## Iteration  5 :   X= 5.13509946189492   Y= 0.00382129831852751
## Convergence failure. Deviation:  0.48003626931828 after  5 iterations
## Iteration  6 :   X= 5.1349367603068   Y= 2.48970923877323e-07
## Convergence failure. Deviation:  0.00382129831852751 after  6 iterations
## Iteration  7 :   X= 5.13493674970484   Y= 1.06581410364015e-14
## Found the root point:  5.13493674970484  after  7 iterations
```

```
## [1] 5.134937
```

```r
abline(h=0,col='red',lwd=1, lty=2)
abline(v=0,col='red',lwd=1, lty=2)
```

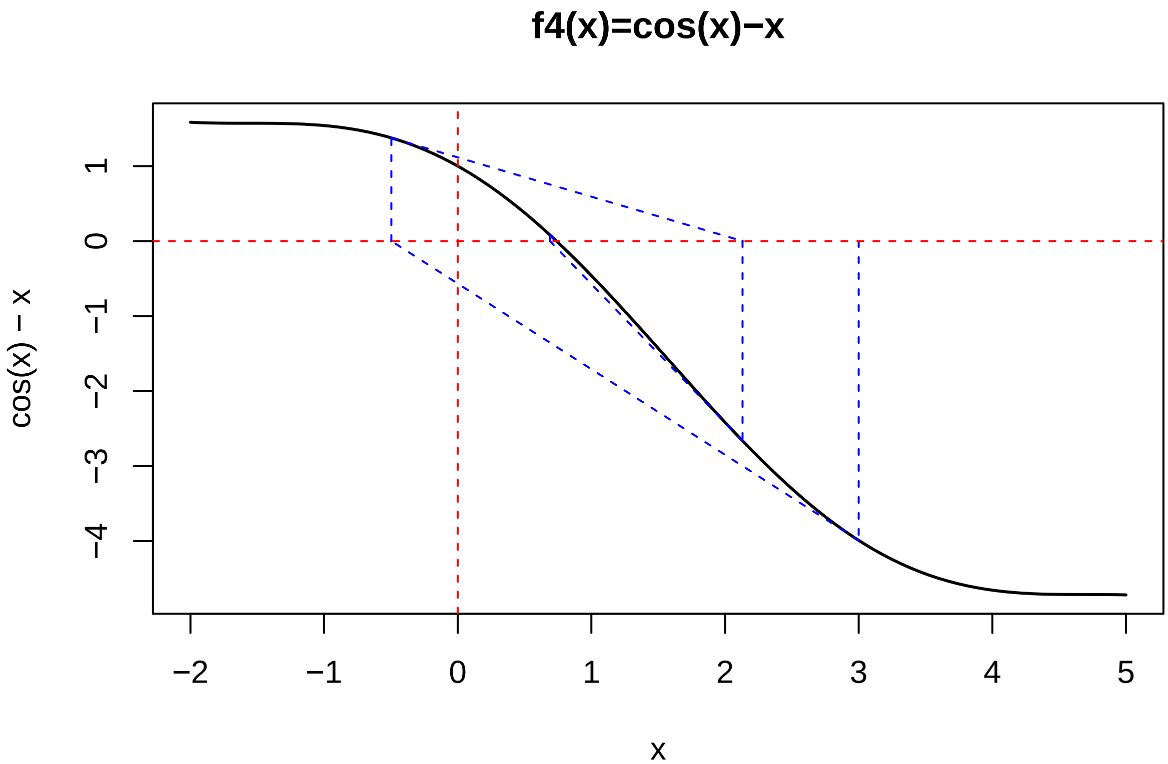



```
# Root of cos(x)-x
curve(cos(x)-x, xlim=c(-2,5), lwd=1.5, main="f4(x)=cos(x)-x")
NewtonRaphson(F4,3,1e-3,40,1)
```

```
##
## Iteration  1 :   X= -0.496558178297331   Y= 1.37578563617707
## Convergence failure. Deviation:  3.98999249660045 after  1 iterations
## Iteration  2 :   X= 2.131003844481   Y= -2.6623658513834
## Convergence failure. Deviation:  1.37578563617707 after  2 iterations
## Iteration  3 :   X= 0.689662720778373   Y= 0.0817979411125979
## Convergence failure. Deviation:  2.6623658513834 after  3 iterations
## Iteration  4 :   X= 0.739652997531334   Y= -0.000950503696277361
## Convergence failure. Deviation:  0.0817979411125979 after  4 iterations
## Iteration  5 :   X= 0.739085204375836   Y= -1.19095364348176e-07
## Found the root point:  0.739085204375836  after  5 iterations
```

```
## [1] 0.7390852
```

```
abline(h=0,col='red',lwd=1, lty=2)
abline(v=0,col='red',lwd=1, lty=2)
```



f4(x)=cos(x)−x
cos(x) − x
1
0
−1
−2
−3
−4
−2
−1
0
1
2
3
4
5
x