markdown prueba

martin

2/9/2021

```
################################################################################
######################### EJERCITACIÓN PARA ENTREGAR############################
################################################################################

#### Ejercicios resueltos con algoritmos del libro Analisis Numerico de Burden & Faires.

### Algoritmos
##Aclaracion
#Definimos una funcion cualquiera, que tenga UNA raiz en el intervalo que queramos estudiar
#La funcion se ira actualizando a medida que se resuelvan ejercicios o se modifique su estructura
#Las tildes seran omitidas para evitar errores (? donde hay letras con tilde)
#Los pasos son exactamente iguales a los señalados en el libro mecionado
#Algunas particularidades han sido agregadas para señalar desbordamiento en el algoritmo de PF


##Biseccion
f <- function(x){x^3+4* x^2- 10}

biseccion <- function(a, b, tol, N){
  #ingresamos extremos de intervalo, tolerancia del error, maximo numero de iteraciones
  #Paso 1
  i=1
  FA <- f(a)

  #Paso 2
  while (i<=N){

    #Paso 3
    p = a + (b-a)/2
    FP = f(p)

    #Paso 4
    if(FP==0 | (b-a)/2 < tol){
        return(p)
    }
    #Paso 5
    i=i+1
    #Paso 6
    if( FA*FP > 0){
      #b no cambia. Se define el nuevo intervalo [p1,b]
      a=p
      FA=FP
```

```r
    } else{
       #El intervalo es [a,p1], FA no cambia
       b=p
    }
  }

  #Pas0 7
  return(paste("El procedimiento falló luego de las",N, "iteraciones especificadas."))
}

biseccion(1,2,10^-4, 12)  #ejemplo del libro para corroborar
```

```
## [1] "El procedimiento falló luego de las 12 iteraciones especificadas."
```

```r
biseccion(1,2,10^-4, 18)  #ejemplo del libro para corroborar
```

```
## [1] 1.365173
```

```r
##Metodo Iteracion de punto fijo
#Se plantea el problema acomodando la funcion para buscar un punto fijo
f <- function(x){0.5*(10-x^3)^0.5}

puntofijo <- function( P0, tol, N){

  i=1

  while(i<=N){
    p=f(P0)
    if(is.na(p)){ return("desbordamiento")}
       else{
         if( abs(p-P0)<tol ){
         return(p)
         }

        i=i+1
        P0=p
       }
  }

  return(paste("El prodcedimiento fallo luego de",N,"iteraciones."))

}

puntofijo(1.5,10^-4, 5)
```

```
## [1] "El prodcedimiento fallo luego de 5 iteraciones."
```

```r
puntofijo(1.5,10^-5, 100)
```

```
## [1] 1.365233
```

```r
## Metodo de Newton-Raphson

# Se define f
f<-function(x){
  x^3+4* x^2- 10

  }
#Debemos crear un objeto que contega la derivada de f para toda x: sea g dicho objeto.
#Instalamos la libreria pertinente
library(Deriv)
#Derivamos f respecto de x, probamos un valor e imprimimos para observar.
g<-Deriv(f, "x")
print(g)
```

```
## function (x)
## x * (3 * x + 8)
```

```r
print(g(1))
```

```
## [1] 11
```

```r
newton <- function(p_inicial, tol, N){
  #paso 1
  i=1

  #paso 2
  while (i<=N){
    #paso 3
    p = p_inicial - f(p_inicial)/g(p_inicial)
    #paso 4
    if(abs(p - p_inicial) <= tol){
        return(p)
      }

    i=i+1
    p_inicial=p
  }

  return(paste("El metodo fallo luego de", N, "iteraciones."))

}
#Probamos con algo ya conocido
newton(1.3, 10^-5, 3)
```

```
## [1] 1.36523
```

```r
## Metodo de la secante.

secante <- function(p_0, p_1, tol, N){
  #paso 1
  i=2
```

```r
  q_0 = f(p_0)
  q_1 = f(p_1)

  #paso 2
  while(i <= N){

    #paso 3
    p = p_1 - (q_1 * (p_1 - p_0)) / (q_1 - q_0)

    #paso 4
    if(abs(p - p_1) <= tol){
      return(p)
    }

    #paso 5
    i = i+1

    #paso 6
     p_0 = p_1
     q_0 = q_1
     p_1 = p
     q_1 = f(p)
  }

  #Paso 7
  return(paste("El procedimiento fallo despues de", N, "iteraciones."))

}
#Nuevamente probamos la funcion con algo conocido. Necesita mas iteraciones
secante(0.5, 1.2, 10^-5, 6)
```

```
## [1] 1.36523
```

```r
## Metodo de posicion falsa

posicion_falsa <- function(p_0, p_1, tol, N){
  #Paso 1
  i = 2
  q_0 = f(p_0)
  q_1 = f(p_1)

  # pAso 2
  while(i <= N){
    #paso 3
    p = p_1 - (q_1 * (p_1 - p_0)) / (q_1 - q_0)

    #paso 4
    if(abs(p - p_1) <= tol){
      return(p)
    }

    #paso 5
    i= i+1
```

```r
    q = f(p)

    #paso 6
    if( q * q_1 < 0){
      p_0 = p_1
      q_0 = q_1
    }

    #paso 7
    p_1 = p
    q_1 = q
  }

  #paso 8
  return(paste("El metodo fallo luego de", N, "iteraciones."))
}

#Probamos.
posicion_falsa(1.67, 1.3, 10^-5, 7)
```

```
## [1] 1.36523
```

```r
#Esta en lo correcto!



### Ejercicios para entregar

##Bisección
f<-function(x){
  cos(x) - x^0.5
}
biseccion(0, 1, 10^-5, 800)
```

```
## [1] 0.641716
```

```r
f<-function(x){
  x^3 + 4*x^2 -10
}
biseccion(1, 2, 10^-4, 17)
```

```
## [1] 1.365173
```

```r
f<-function(x){
  2 + cos( exp(x) -2) - exp(x)
}
biseccion(0, 4, 10^-5, 120)
```

```
## [1] 1.007622
```

```r
f<-function(x){
  x^3 - 7*x^2 + 14*x - 6
}
biseccion(-5, 5, 10^-5, 20)
```

```
## [1] 0.5857944
```

```r
##Punto fijo
f<-function(x){
 x - x^3 - 4*x^2 + 10
}
puntofijo(1.2, 10^-4, 30)
```

```
## [1] "desbordamiento"
```

```r
f<-function(x){
 (10/x - 4*x)^0.5
}
puntofijo(1.36 , 10^-4, 40)
```

```
## [1] "desbordamiento"
```

```r
f<-function(x){
  0.5*(10-x^3)^0.5
}
puntofijo(1.5, 10^-5, 50)
```

```
## [1] 1.365233
```

```r
f<-function(x){
  (10/(4+x))^0.5
}
puntofijo(1.5, 10^-5, 100)
```

```
## [1] 1.365231
```

```r
f<-function(x){
 x-( (x^3+4*x^2-10)/(3*x^2+8*x))
}

puntofijo(1.5, 10^-5, 100)
```

```
## [1] 1.36523
```

```r
#Problemas rescritos como problemas de punto fijo para hallar raices
f<-function(x){
  ( cos(x) )^0.5
}

puntofijo(0, 10^-5, 211)
```

```
## [1] 0.8241346
```

```r
puntofijo(-1, 10^-4, 1000)
```

```
## [1] 0.8241063
```

```r
f<-function(x){
  log( 2 + cos( exp(x)-2) )
}
puntofijo(8, 10^-4, 500)
```

```
## [1] 1.007589
```

```r
puntofijo(400, 10^-12, 666)
```

```
## [1] 1.007624
```

```r
f<-function(x){
  -7 + 14/x- 6/x^2
}

puntofijo(0, 10^-4, 500)
```

```
## [1] "desbordamiento"
```

```r
puntofijo(8, 10^-4,500)
```

```
## [1] -8.690401
```

```r
puntofijo(5000, 10^-211, 666)
```

```
## [1] -8.690416
```

```r
##Newton, secante y falsa posición
#a
f<-function(x){
  exp(x) + 2^(-x) + 2*cos(x) - 6
}
biseccion(1, 2, 10^-4, 200)
```

```
## [1] 1.829407
```

```r
g <- Deriv(f,"x")
newton(biseccion(1, 2, 10^-4, 1000), 10^-4, 30)
```

```
## [1] 1.829384
```

```r
secante(1.7, 1.829407, 10^-4, 60)
```

```
## [1] 1.829381
```

```r
posicion_falsa(1.7, 1.892407, 10^-4, 90)
```

```
## [1] 1.829383
```

```r
#b
f<-function(x){
  log(x-1) + cos(x-1)
}
g<-Deriv(f,"x")
g
```

```
## function (x)
## {
##     .e1 <- x - 1
##     1/.e1 - sin(.e1)
## }
```

```r
biseccion(1, 2, 10^-4, 1000)
```

```
## [1] 1.397766
```

```r
newton(1.397766 , 10^-4, 30)
```

```
## [1] 1.397748
```

```r
secante(1, 1.397766, 10^-4, 60)
```

```
## [1] 1.397766
```

```r
posicion_falsa(1, 1.397766, 10^-4, 90)
```

```
## [1] 1.397766
```

```r
#c
f<-function(x){
  2*x*cos(2*x) -(x-2)^2
}
g<-Deriv(f,"x")
g
```

```
## function (x)
## {
##     .e1 <- 2 * x
##     2 * cos(.e1) - (2 * (x - 2) + 4 * (x * sin(.e1)))
## }
```

```r
#Para [2,3]
biseccion(2,3, 10^-4, 1000)
```

```
## [1] 2.370667
```

```r
newton(2.370667, 10^-4, 30)
```

```
## [1] 2.370687
```

```r
secante(0, 2.370667, 10^-4, 60)
```

```
## [1] 2.370687
```

```r
posicion_falsa(0, 2.370667, 10^-4, 60)
```

```
## [1] 2.370687
```

```r
#Para [3,4]
biseccion(3,4, 10^-4, 1000)
```

```
## [1] 3.722107
```

```r
newton(3.722107, 10^-4, 30)
```

```
## [1] 3.722113
```

```r
secante(-30, 3.72, 10^-4, 100)
```

```
## [1] 3.722113
```

```r
posicion_falsa(0, 4, 10^-4, 1000) #falla ¿por que?
```

```
## [1] 2.370686
```

```r
posicion_falsa(3, 3.72, 10^-4, 1000) #¿necesita de dos muy buenas aproximaciones?
```

```
## [1] 3.722113
```

```r
#d
f<-function(x){
  (x - 2)^2 - log(x)
}
g<-Deriv(f,"x")
g
```

```
## function (x)
## 2 * (x - 2) - 1/x
```

```r
#Para [1,2]
biseccion(1,2, 10^-4, 1000)
```

```
## [1] 1.412415
```

```r
newton(1.412415, 10^-4, 100)
```

```
## [1] 1.412391
```

```r
secante(1, 1.42, 10^-4, 150)
```

```
## [1] 1.412391
```

```r
posicion_falsa(1, 1.42, 10^-4, 200)
```

```
## [1] 1.41241
```

```r
#Para [e,4]
biseccion(exp(1), 4, 10^-4, 500)
```

```
## [1] 3.057095
```

```r
newton(3.057095, 10^-4, 30)
```

```
## [1] 3.057104
```

```r
secante(3.57, 3.057095, 10^-4, 200)
```

```
## [1] 3.057102
```

```r
posicion_falsa(3, 3.57095, 10^-4, 250)
```

```
## [1] 3.057094
```

```r
#e
f<-function(x){
  exp(x) - 3*x^2
}
g<-Deriv(f,"x")
g
```

```
## function (x)
## exp(x) - 6 * x
```

```r
#Para [0,1]
biseccion(0, 1, 10-6, 1000)
```

```
## [1] 0.5
```

```r
newton(0.5, 10^-4, 9000)
```

```
## [1] 0.9100076
```

```r
secante(0.6, 0.5, 10^-4, 200)
```

```
## [1] 0.9100076
```

```r
posicion_falsa(1, 0.5, 10^-4, 250)
```

```
## [1] 0.9100039
```

```r
#Para [3,5]
biseccion(3,5, 10^-4, 1000)
```

```
## [1] 3.733093
```

```r
newton(3.733093, 10^-4, 100)
```

```
## [1] 3.733079
```

```r
secante(3.73, 3.733093, 10^-4, 200)
```

```
## [1] 3.733079
```

```r
posicion_falsa(3, 3.73, 10^-4, 250)
```

```
## [1] 3.733079
```

```r
#f
f<-function(x){
  sin(x) - exp(-x)
}
g<-Deriv(f,"x")
g
```

```
## function (x)
## cos(x) + exp(-x)
```

```r
#Para [0,1]
biseccion(0,1, 10^-4, 500)
```

```
## [1] 0.588562
```

```r
newton(0.58853, 10^-4, 100)
```

```
## [1] 0.5885327
```

```r
secante(0.58, 0.58853, 10^-4, 200)
```

```
## [1] 0.5885327
```

```r
posicion_falsa(3, 0.5883, 10^-4, 250)
```

```
## [1] 0.5885327
```

```r
#Para [3,4]
biseccion(3,4, 10^-4, 500)
```

```
## [1] 3.096375
```

```r
newton(3.0963, 10^-4, 100)
```

```
## [1] 3.096364
```

```r
secante(3.57, 3.096375, 10^-4, 200)
```

```
## [1] 3.096364
```

```r
posicion_falsa(3, 3.1, 10^-4, 250)
```

```
## [1] 3.096364
```

```r
#Para [6,7]
biseccion(6,7, 10^-4, 500)
```

```
## [1] 6.285095
```

```r
newton(6.2850, 10^-4, 200)
```

```
## [1] 6.285049
```

```r
secante(3, 6.285, 10^-4, 200)
```

```
## [1] 6.285049
```

```r
posicion_falsa(3, 6.285, 10^-4, 250)
```

```
## [1] 3.096364
```

```r
##Otras funciones

f<-function(x){
  (cos(x))^0.5
}
#Aproximacion inicial por PF
puntofijo(0, 10^-4, 1000)
```

```
## [1] 0.8241063
```

```r
f<-function(x){
  cos(x) - x^0.5
}
g<-Deriv(f,"x")
g
```

```
## function (x)
## -(0.5/sqrt(x) + sin(x))
```

```r
newton(0.8241063, 10^-4, 100)
```

```
## [1] 0.6417144
```

```r
secante(1, 0.8241, 10^-4, 200)
```

```
## [1] 0.6417144
```

```r
posicion_falsa(1, 0.8241, 10^-4, 250)
```

```
## [1] 0.6417152
```

```r
f<-function(x){
  log(2+cos(exp(x) - 2))
}
puntofijo(2, 10^-4, 1000)
```

```
## [1] 1.007592
```

```r
f<-function(x){
  2 + cos(exp(x)-2) - exp(x)
}
g<-Deriv(f,"x")
newton(1.0075, 10^-4, 100)
```

```
## [1] 1.007624
```

```r
secante(0.5, 1.007624, 10^-4, 200)
```

```
## [1] 1.007624
```

```r
posicion_falsa(0.5, 0.007624, 10^-4, 250)
```

```
## [1] 1.007562
```

```r
f<-function(x){
  x^3-7*x^2+14*x-6
}
g<-Deriv(f,"x")
biseccion(0,10, 10^-4, 1000)
```

```
## [1] 0.5858612
```

```r
newton(0.5858612, 10^-10, 300)
```

```
## [1] 0.5857864
```

```r
secante(0.5, 0.5858, 10^-4, 200)
```

```
## [1] 0.5857873
```

```r
posicion_falsa(0.5, 0.5858, 10^-4, 250)
```

```
## [1] 0.5857873
```

```r
f<-function(x){
  cos(x)
}
puntofijo(1, 10^-70, 1000)
```

```
## [1] 0.7390851
```

```r
f<-function(x){
  cos(x)-x
}
g<-Deriv(f,"x")
newton(0.7390851, 10^-4, 100)
```

```
## [1] 0.7390851
```

```r
secante(0.1, 0.739, 10^-4, 200)
```

```
## [1] 0.7390851
```

```r
posicion_falsa(0.1, 0.739, 10^-4, 250)
```

```
## [1] 0.7390851
```

```r
f<-function(x){
  -x^3 + cos(x)
}
biseccion(-1, 1, 10^-4, 1000)
```

```
## [1] 0.8654175
```

```r
g<-Deriv(f,"x")
newton(0.865474, 10^-4, 100)
```

```
## [1] 0.865474
```

```r
secante(8, 0.865474, 10^-4, 200)
```

```
## [1] 0.865474
```

```r
posicion_falsa(8, 0.865474, 10^-4, 250)
```

```
## [1] 0.865474
```

```r
################################################################################
################################################################################
```