

SEnoL Newton

Agustin Huczok

30/9/2021

*##*Dos variables *##*Norma

```
norma <- function(y, metodo){  
  if (metodo==2){  
    return(sqrt(sum(y^2)))  
  }  
  if (metodo==Inf){  
    return(max(abs(y)))  
  }  
  return("El metodo debe ser 2 o Inf")  
}
```

*##*Algoritmo

```
Newton_SEnoL<- function(n,x,TOL,N){  
  #Paso 1  
  k <- 1  
  #Paso 2  
  while(k<=N){  
    #Paso 3  
    fx <- Fx(x)  
    J <- Jacobiano(x[1],x[2])  
    #Paso 4  
    y = solve(J)%*%-fx  
    #Paso 5  
    x <- x + t(y)  
    #Paso 6  
    if (norma(y,2) < TOL){  
      return(x)  
    }  
    #Paso 7  
    k <- k+1  
  }  
  #Paso 8  
  return(paste('Numero max de iteraciones excedido'))  
}
```

*##*Calculo derivadas dos variables

```
fa=function(x1,x2){  
}
```

```
fae=expression()  
D(fae,"x1")
```

```
## [1] NA
```

```
D(fae,"x2")
```

```
## [1] NA
```

```
dfa1=function(x1,x2){}  
dfa2=function(x1,x2){}
```

```
fb=function(x1,x2){  
}  
fbe=expression()  
D(fbe,"x1")
```

```
## [1] NA
```

```
D(fbe,"x2")
```

```
## [1] NA
```

```
dfb1=function(x1,x2){}  
dfb2=function(x1,x2){}
```

```
##Matriz jacobiana
```

```
Jacobiano <- function(x1,x2){  
  col1 <-  
    c(dfa1(x1,x2),dfa2(x1,x2))  
  
  col2 <-  
    c(df b1(x1,x2),df b2(x1,x2))  
  
  J <- rbind(col1,col2) #armo la matriz ampliada  
  return(J)  
}
```

```
##Definino Fx
```

```

Fx <- function(x){
  Fx <- rbind(fa(x[1],x[2]), fb(x[1],x[2]))
  return(Fx)
} #sera una matriz ampliada con las funciones definadas antes

```

Evaluo fn y el Jacobiano

```

x <- c(0,0)
n=2
#Sist_Ec_NoLineal_Newton(n, x, 10^-6, 100)

```

Corroboro

```

#Asigno los rdos del algoritmo a las variables x1,x2
#x1 <- NewtonSEnoL(n,x, 10^-5, 100)[1] #posicion, osea mult por posicion 1
#x2 <- NewtonSEnoL(n,x, 10^-5, 100)[2]

```

##Resultados

```

#fa(x1, x2)
#fb(x1, x2)

```

#Tres variables ##Norma

```

norma <- function(y, metodo){
  if (metodo==2){
    return(sqrt(sum(y^2)))
  }
  if (metodo==Inf){
    return(max(abs(y)))
  }
  return("El metodo debe ser 2 o Inf")
}

```

##Sistema Ec. No lineal Newton

```

NewtonSEnoL <- function(n,x,TOL,N){
  #Paso 1
  k <- 1
  #Paso 2
  while(k<=N){
    #Paso 3
    fx <- Fx(x)
    J <- Jacobiano(x[1],x[2],x[3])
    #Paso 4
    y = solve(J)%*%-fx
    #Paso 5
    x <- x + t(y)
  }
}

```

```

#Paso 6
if (norma(y,2) < TOL){
  return(x)
}
#Paso 7
k <- k+1
}
#Paso 8
return(paste('Numero max de iteraciones excedido'))
}

```

##Calculo derivadas tres variables

```

fa=function(x1,x2,x3){
  5*x1+2*x2
}
fae=expression(5*x1+2*x2)
D(fae,"x1")

```

```
## [1] 5
```

```
D(fae,"x2")
```

```
## [1] 2
```

```
D(fae,"x3")
```

```
## [1] 0
```

```

dfa1=function(x1,x2,x3){}
dfa2=function(x1,x2,x3){}
dfa3=function(x1,x2,x3){}

fb=function(x1,x2,x3){

}

fbe=expression()
D(fbe,"x1")

```

```
## [1] NA
```

```
D(fbe,"x2")
```

```
## [1] NA
```

```
D(fbe,"x3")
```

```
## [1] NA
```

```
dfb1=function(x1,x2,x3){}
dfb2=function(x1,x2,x3){}
dfb3=function(x1,x2,x3){}
```

```
fc=function(x1,x2,x3){
}
```

```
fce=expression()
D(fce,"x1")
```

```
## [1] NA
```

```
D(fce,"x2")
```

```
## [1] NA
```

```
D(fce,"x3")
```

```
## [1] NA
```

```
dfc1=function(x1,x2,x3){}
dfc2=function(x1,x2,x3){}
dfc3=function(x1,x2,x3){}
```

```
##Matriz Jacobiana
```

```
Jacobiano <- function(x1,x2,x3){
  col1 <-
    c(dfa1(x1,x2,x3),dfa2(x1,x2,x3),dfa3(x1,x2,x3))

  col2 <-
    c(dfb1(x1,x2,x3),dfb2(x1,x2,x3),dfb3(x1,x2,x3))

  col3 <-
    c(dfc1(x1,x2,x3),dfc2(x1,x2,x3),dfc3(x1,x2,x3))

  J <- rbind(col1,col2, col3) #con esta ultima armamos la matrix ampliada
  return(J)
}
```

```
##Defino Fx
```

```
Fx <- function(x){
  Fx <- rbind(fa(x[1],x[2],x[3]), fb(x[1],x[2],x[3]), fc(x[1],x[2],x[3]))
  return(Fx)
} #sera una matriz ampliada con las funciones definadas antes
```

##Defino los puntos

```
x <- c(0.1, 0.1, -0.1)
n=3
#Resultado <- Sist_Ec_NoLineal_Newton(n, x, 10^-6, 100)
#Resultado
```

##Corroboro

```
#x1 <- NewtonSEnoL(n,x, 10^-6, 100)[1] #posicion, osea mult por posicion 1
#x2 <- NewtonSEnoL(n,x, 10^-6, 100)[2]
#x3 <- NewtonSEnoL(n,x, 10^-6, 100)[3]
```

##Imprimo los resultados

```
#fa(x1, x2, x3)
#fb(x1, x2, x3)
#fc(x1, x2, x3)
```