Ejercicios

Análisis Numérico

Joaquin Cavieres G.

Método de Newton

El método de Newton (o también conocido como Newton-Raphson) es uno de los métodos numéricos más conocidos y utilizados para encontrar la raíz de una función. Además, es uno de los métodos con mayor velocidad de convergencia en comparación con otros métodos numéricos.

- Partimos de un valor inicial de x_0
- Se comienza el algoritmo e iterativamente se va estimando un nuevo valor mediante:

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

Desarrollo paso a paso

Suponga que deseamos encontrar la raíz de una función $f(x)$, lo que en terminos práticos singifica encontrar un nuevo valor x^* tal que $f(x^*) = 0$. Por tanto, el método de Newton usualmente nos permite encontrar iterativamente ese x^* mediante calculos sencillos.

Desarrollo paso a paso:

- Damos un valor inicial x_0.
- Aproximamos linealmente a la función $f(x)$ alrededor de x_0.
- Encontramos la raíz de esta aproximación.
- Iterar "n"pasos hasta que el método converga

La aproximación lineal alrededor de $x = x_0$ esta determinada por:

$$f(x) \approx f(x_0) + f^{'}(x_0)(x - x_0)$$

La solución de $f(x) = 0$ no es tan simple, por lo que el método de Newton nos ayuda a encontrar la raíz de la aproximación lineal mediante:

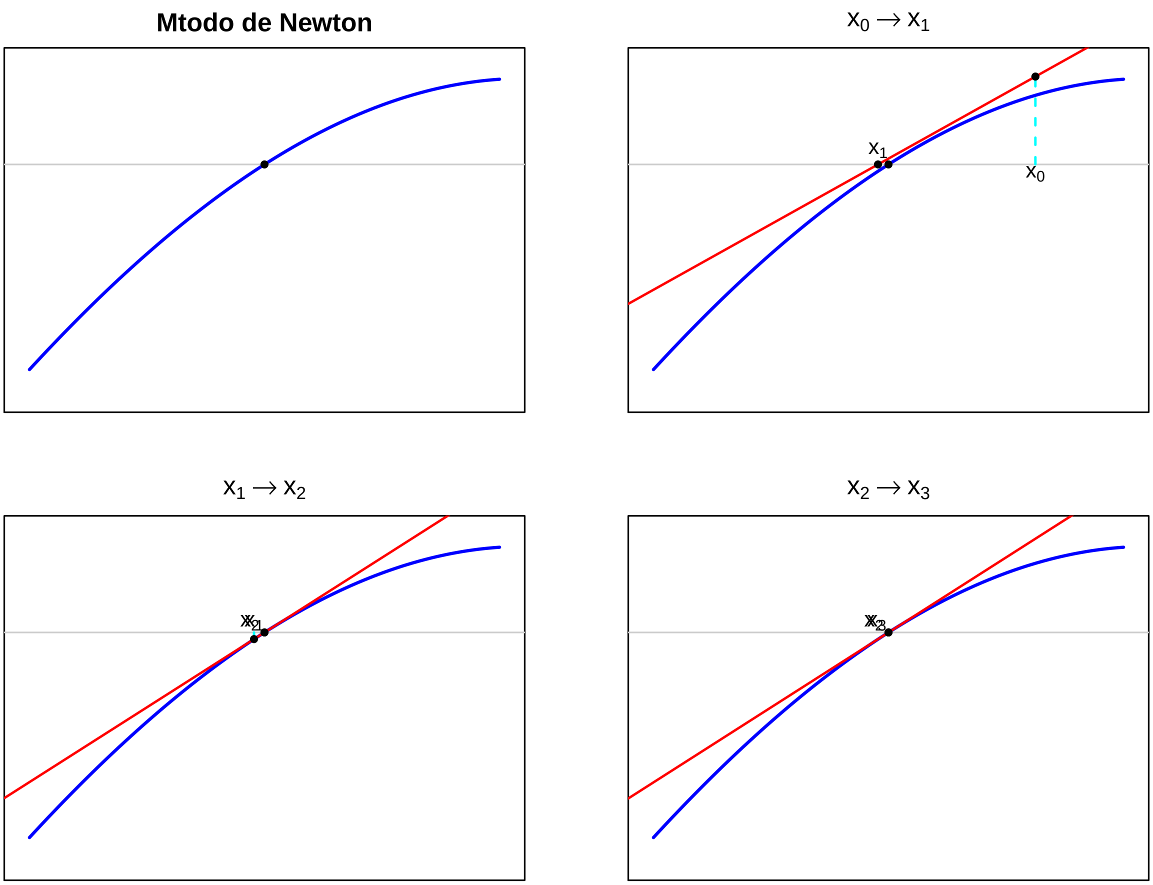
$$x_1 = x_0 - \frac{f(x_0)}{f'(x_0)}$$

Considerando ahora a x_1 como nuestro nuevo punto inicial entonces seguimos iterando dentro del algoritmo.

Ejemplo: Encuentre la solución para la función: $f(x) = x^4 - x^3 + 2x$ y punto inicial $x_0 = 0{,}5$

```r
# Creamos la función
f = function(x){x^4 - x^3 + 2*x}

# Su derivada
fp = function(x){4*x^3 - 3*x^2 + 2}

# Creamos los gráficos para las soluciones paso a paso
par(mfrow=c(2,2),mar=c(2,2,2,2))
xx <- seq(-.8,.8,length=200)
yy <- 2*xx-xx^2-.1*xx^3
plot(xx,yy,type="l",xlab="",ylab="",xlim=c(-.82,.82),ylim=c(-2.5,1.1),
     col="blue",lwd=2,xaxt="n",yaxt="n", main=c("Mtodo de Newton"))
abline(h=0,col=gray(.8))
points(0,0,pch=20)

x0 <- 0.5
x1 <- x0 - f(x0)/fp(x0)
plot(xx,yy,type="l",xlab="",ylab="",xlim=c(-.82,.82),ylim=c(-2.5,1.1),
     main=expression(x[0] %->% x[1]),col="blue",lwd=2,xaxt="n",yaxt="n")
abline(h=0,col=gray(.8))
abline(a=f(x0)-fp(x0)*x0, b=fp(x0), col="red",lwd=1.5)
segments(x0,0,x0,f(x0),lty=2,col="cyan",lwd=1.5)
points(c(x0,x1,0),c(f(x0),0,0),pch=20)
text(x0,-.1,expression(x[0]))
text(x1,.15,expression(x[1]))


x0 <- x1
x1 <- x0 - f(x0)/fp(x0)
plot(xx,yy,type="l",xlab="",ylab="",xlim=c(-.82,.82),ylim=c(-2.5,1.1),
     main=expression(x[1] %->% x[2]),col="blue",lwd=2,xaxt="n",yaxt="n")
abline(h=0,col=gray(.8))
abline(a=f(x0)-fp(x0)*x0, b=fp(x0), col="red",lwd=1.5)
segments(x0,0,x0,f(x0),lty=2,col="cyan",lwd=1.5)
points(c(x0,x1,0),c(f(x0),0,0),pch=20)
text(x0,.1,expression(x[1]))
text(x1-.05,.1,expression(x[2]))
```

```
x0 <- x1
x1 <- x0 - f(x0)/fp(x0)
plot(xx,yy,type="l",xlab="",ylab="",xlim=c(-.82,.82),ylim=c(-2.5,1.1),
     main=expression(x[2] %->% x[3]),col="blue",lwd=2,xaxt="n",yaxt="n")
abline(h=0,col=gray(.8))
abline(a=f(x0)-fp(x0)*x0, b=fp(x0), col="red",lwd=1.5)
segments(x0,0,x0,f(x0),lty=2,col="cyan",lwd=1.5)
points(c(x0,x1,0),c(f(x0),0,0),pch=20)
text(x0-.05,.1,expression(x[2]))
text(x1-.04,.1,expression(x[3]))
```



El algoritmo tiene la siguiente estructura:

- Iniciar con un valor de x_0.
- Para un $n = 1, ..., $ hacer:

$$x_n = x_{n-1} - \frac{f(x_{n-1})}{f'(x_{n-1)})}$$

- Parar el algoritmo si $|x_n - x_{n-1}| < \delta$. Donde δ es un valor de tolerancia que fijamos inicialmente.

El siguiente algoritmo es el muismo método pero optimizado en una `function` escrita en `R`.

```
# Creamos la función
newton = function(f, a, b, tol = 1e-5, n = 1000) {
  require(numDeriv) # Librería para calcular la derivada de f(x)

  x0 = a # Valor inicial para el intervalo inferior
  k  = n # índice para el número de tteraciones

  # Verificicar si las aproximaciones dan como resultado 0
  fa = f(a)
  if (fa == 0.0) {
    return(a)
  }

  fb = f(b)
  if (fb == 0.0) {
    return(b)
  }

  # Partir con el proceso iterativo
  for (i in 1:n) {
    dx = genD(func = f, x = x0)$D[1] # Derivada de primer orden f'(x)
    x1 = x0 - (f(x0) / dx) # Calcular x1
    k[i] = x1 # Guardar x1
    # Una vez que la diferencia entre x0 y x1 sea lo suficientemente
    # pequeña, genere los resultados.
    if (abs(x1 - x0) < tol) {
      root.approx <- tail(k, n = 1)
      res <- list('Approximacion' = root.approx, 'Iteraciones' = k)
      return(res)
    }
    # Si el método de Newton aún no ha alcanzado la convergencia,
    # establezca x1 como x0 y continúe
    x0 = x1
  }
  print('No se encontro la solucion en el numero de iteraciones dadas')
}
```

Ya creado el algoritmo escribimos nuestra función:

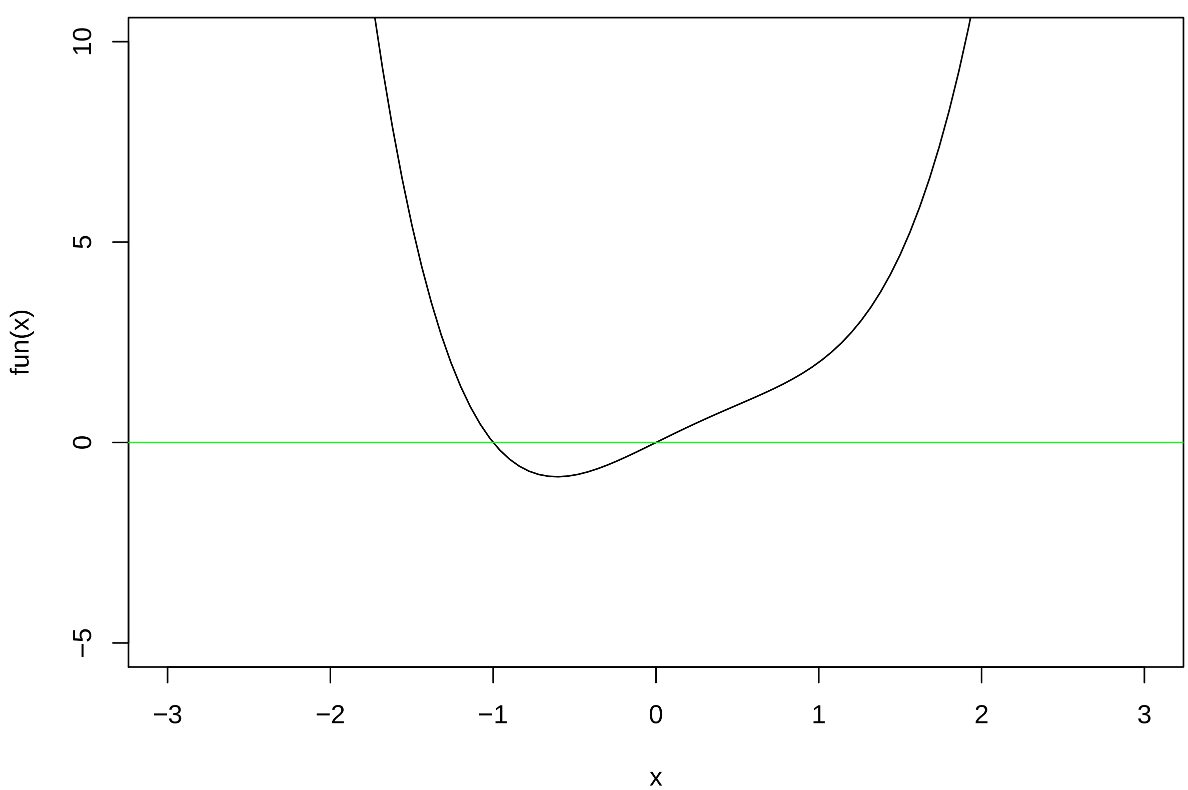
```
f = function(x){x^4 - x^3 + 2*x}
```

y aplicamos el método:

```r
newton(f, -5, 5, tol = 1e-6, n = 100)
```

```
## $Approximacion
## [1] -1
##
## $Iteraciones
##  [1] -3.708551 -2.751866 -2.053503 -1.560340 -1.238100 -1.064263 -1.006411
##  [8] -1.000073 -1.000000 -1.000000
```

```r
## Grafico
fun = function(x){x^4 - x^3 + 2*x}
curve(fun, -3, 3, 101, ylim = c(-5,10))
abline(0,0,col="green")
```



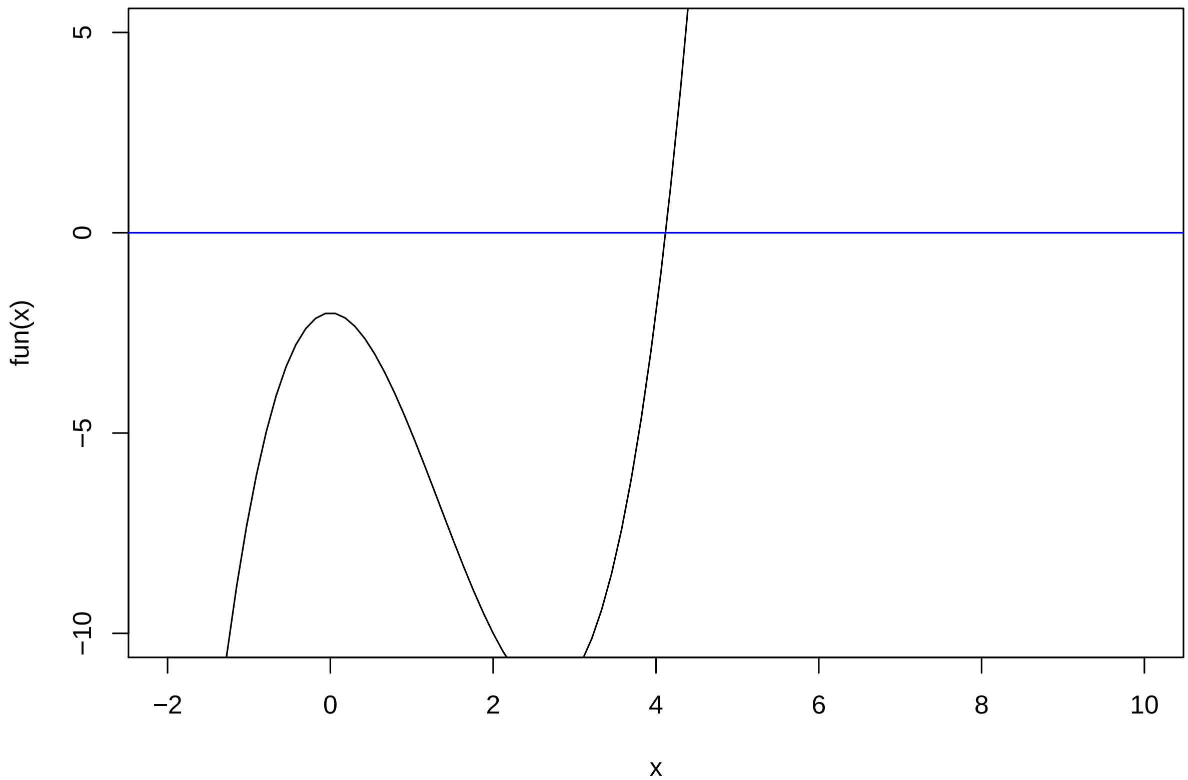
Ejemplo 2: Encuentre la raíz de $f(x) = x^3 - 4x^2 - 2$.

```
f2 = function(x){x^3 - 4*x^2 -2}
```

```
newton(f2, -5, 5, tol = 1e-6, n = 100)
```

```
## $Approximacion
## [1] 4.117942
##
## $Iteraciones
##  [1]  -3.02608696  -1.74243929  -0.89920954  -0.27948419   0.66548757
##  [6]  -0.20473138   1.02924810   0.01122153 -22.36714851 -14.51314065
## [11]  -9.29729596  -5.84682234  -3.57925664  -2.10165925  -1.13869112
## [16]  -0.47228120   0.20171741  -1.24267207  -0.54994134   0.08623112
## [21]  -2.95343291  -1.69523008  -0.86726411  -0.25158462   0.77862650
## [26]  -0.11769490   1.97467237  -0.43969470   0.25787609  -0.94890754
## [31]  -0.32164502   0.52700258  -0.34935405   0.45129427  -0.45648521
## [36]   0.22825282  -1.08723700  -0.43275762   0.27058098  -0.89807678
## [41]  -0.27850623   0.66911877  -0.20157560   1.04991920   0.01857797
## [46] -13.54193989  -8.65359895  -5.42251198  -3.30187983  -1.92155240
## [51]  -1.01928523  -0.37917536   0.37979154  -0.58819184   0.03641168
## [56]  -6.94285525  -4.29746781  -2.56844265  -1.44466158  -0.69469303
## [61]  -0.08577873   2.78031224  14.83667394  10.43663831   7.56295844
## [66]   5.74647765   4.69792245   4.22972843   4.12334635   4.11795583
## [71]   4.11794227   4.11794227
```

```
## Grafica
fun <- function(x)(x^3 - 4*x^2 -2)
curve(fun, -2, 10, 100,  ylim = c(-10, 5))
abline(0 , 0, col = "blue")
```



Ejemplo 3

Encontrar la raíz de: $f(x) = x^2 - 2$ y punto de partida $= 2$.

Solución:

Ya que $f(x) = x^2 - 2$ y su derivada es $f'(x) = 2x$, entonces:

Primera iteración del método:

$$x_1 = x_0 - \frac{x^2 - 2}{2x} = 2 - \frac{(2)^2 - 2}{2(2)} = 2 - \frac{(2^2 - 2)}{2 * 2} = 1{,}5$$

Segunda iteración del método:

$$x_2 = x_1 - \frac{x^2 - 2}{2x} = 1{,}5 - \frac{(1{,}5)^2 - 2}{2(1{,}5)} = 1{,}5 - \frac{(1{,}5^2 - 2)}{1{,}5 * 2} = 1{,}41$$

Tercera iteración del método:

$$x_3 = x_2 - \frac{x^2 - 2}{2x} = 1{,}41 - \frac{(1{,}41)^2 - 2}{2(1{,}41)} = 1{,}41 - \frac{(1{,}41^2 - 2)}{1{,}41 * 2} = 1{,}41422$$

. . . .

$\approx$ 1.414214