

Método de bisección

Juan Anagrita, Hector Hernandez, Aldemar Ramirez

Agosto 3, 2019

Problema

Hayar la raíz de una función en un rango $[a,b]$ a través del método de bisección.

Solución

Lenguaje de programación: R

Función principal-método de bisección

Parametros:

-f <- funcion

-xa <- a en el rango $[a,b]$ donde se busca la raíz

-xb <- b en el rango $[a,b]$ donde se busca la raíz

- tol <- tolerancia minima que debe tener la funcion

Valores de retorno:

-a_c <- valor de a traves de las iteraciones

-b_c <- valor de b traves de las iteraciones

-m_c <- valor de m traves de las iteraciones

-i <- iteraciones necesarias para llegar a la tolerancia

-dx <- Error estimado.

```
biseccion = function(f, xa, xb, tol){  
  if( sign(f(xa)) == sign(f(xb)) ){ stop("f(xa) y f(xb) tienen el mismo signo") }  
  a = min(xa,xb)  
  b = max(xa,xb)  
  k = 0  
  
  #k es el numero de iteraciones  
  a_c <- c()  
  b_c <- c()  
  m_c <- c()  
  dx <- c()  
  errorRelativo = c()  
  repeat{  
    m = a + 0.5*(b-a)  
    m_c = c(m_c, m)
```

```

if( f(m)==0 ){
  lista = list("a_c" = a_c, "b_c" = b_c, "m_c" = m_c, "i" = k, "resultado" = m,
              "dx" = dx)
  return(lista)
}

if( sign(f(a)) != sign(f(m)) ){
  b = m
} else { a = m }

a_c = c(a_c,a)
b_c = c(b_c,b)

dx = c(dx,(b-a)/2) #error estimado
k = k+1

#until
if( dx[k] < tol ){
  lista = list("a_c" = a_c, "b_c" = b_c, "m_c" = m_c, "i" = k, "resultado" = m,
              "dx" = dx)
  return(lista)
  break;
}
} #repeat
}

```

La función básicamente toma un intervalo $[a,b]$ de la función, tal que $f(a)*f(b)<0$, y basándose en el teorema de los valores intermedios se sabe que al ser la función continua hay al menos una raíz de la función en este intervalo.

Implementacion

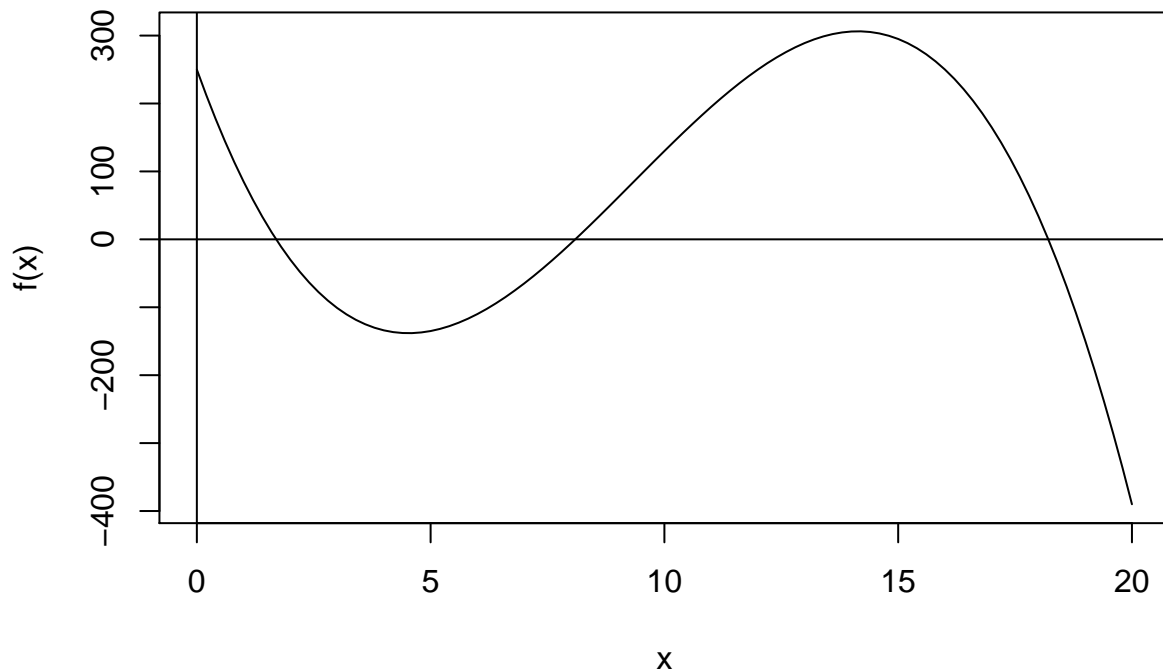
Gráfica de la función

```

f = function(x) (-x)^3+28*x^2-192*x+250
curve(f, 0,20); abline(h=0, v=0) #gráfico para decidir un intervalo
title(main="y = (-x)^3+28*(x^2)-192*x+250")

```

$$y = (-x)^3 + 28(x^2) - 192x + 250$$



A través de la gráfica se puede ver que la función f tiene 3 raíces.

Resultados de las raíces

```
resultados <- biseccion(f, 0, 5, 1e-7)
```

```
cat("Cero de f en [0,5] es approx: ",resultados$resultado, "con error <=",
    resultados$dx[resultados$i],"\n")
```

```
## Cero de f en [0,5] es approx: 1.696277 con error <= 7.450581e-08
```

```
resultados <- biseccion(f, 5, 10, 1e-7)
```

```
cat("Cero de f en [5,10] es approx: ",resultados$resultado, "con error <=",
    resultados$dx[resultados$i],"\n")
```

```
## Cero de f en [5,10] es approx: 8.09322 con error <= 7.450581e-08
```

```
resultados <- biseccion(f, 15, 20, 1e-7)
```

```
cat("Cero de f en [15,20] es approx: ",resultados$resultado, "con error <=",
    resultados$dx[resultados$i],"\n")
```

```
## Cero de f en [15,20] es approx: 18.2105 con error <= 7.450581e-08
```

Tabla de resultados a traves de las iteraciones: para el rango [15,20]

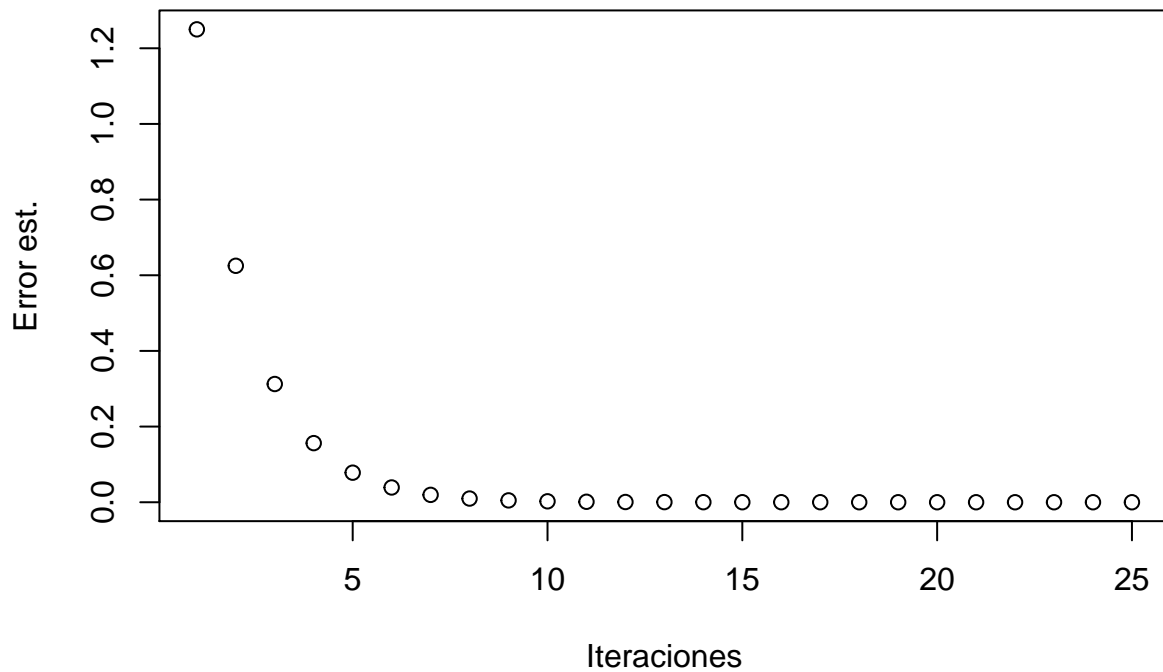
```
tabla <- data.frame(  
  "iteraciones" = 1:resultados$i,  
  "a" = resultados$a_c,  
  "b" = resultados$b_c,  
  "m" = resultados$m_c,  
  "Error est." = resultados$dx  
)  
  
print(tabla)
```

##	iteraciones	a	b	m	Error.est.
## 1	1	17.50000	20.00000	17.50000	1.250000e+00
## 2	2	17.50000	18.75000	18.75000	6.250000e-01
## 3	3	18.12500	18.75000	18.12500	3.125000e-01
## 4	4	18.12500	18.43750	18.43750	1.562500e-01
## 5	5	18.12500	18.28125	18.28125	7.812500e-02
## 6	6	18.20312	18.28125	18.20312	3.906250e-02
## 7	7	18.20312	18.24219	18.24219	1.953125e-02
## 8	8	18.20312	18.22266	18.22266	9.765625e-03
## 9	9	18.20312	18.21289	18.21289	4.882812e-03
## 10	10	18.20801	18.21289	18.20801	2.441406e-03
## 11	11	18.21045	18.21289	18.21045	1.220703e-03
## 12	12	18.21045	18.21167	18.21167	6.103516e-04
## 13	13	18.21045	18.21106	18.21106	3.051758e-04
## 14	14	18.21045	18.21075	18.21075	1.525879e-04
## 15	15	18.21045	18.21060	18.21060	7.629395e-05
## 16	16	18.21045	18.21053	18.21053	3.814697e-05
## 17	17	18.21049	18.21053	18.21049	1.907349e-05
## 18	18	18.21049	18.21051	18.21051	9.536743e-06
## 19	19	18.21050	18.21051	18.21050	4.768372e-06
## 20	20	18.21050	18.21051	18.21050	2.384186e-06
## 21	21	18.21050	18.21050	18.21050	1.192093e-06
## 22	22	18.21050	18.21050	18.21050	5.960464e-07
## 23	23	18.21050	18.21050	18.21050	2.980232e-07
## 24	24	18.21050	18.21050	18.21050	1.490116e-07
## 25	25	18.21050	18.21050	18.21050	7.450581e-08

Grafica de iteraciones vs error estimado

```
plot(x = 1:length(resultados$a_c), y = resultados$dx,  
     xlab = "Iteraciones", ylab = "Error est.",  
     main = "iteraciones vs error estimado")
```

iteraciones vs error estimado



El error estimado siempre converge a cero.

Error estimado : $b - a/(2 * k)$

donde k es la numero de la iteracion

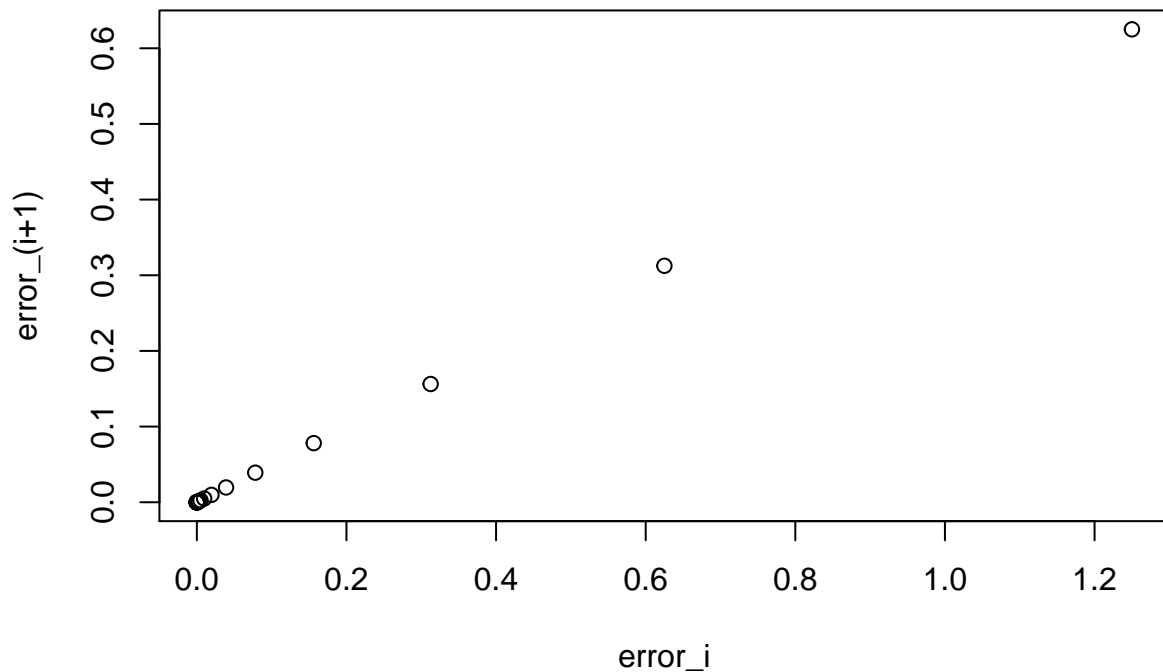
cuando k tiene a infinito el error estimado tiene a 0.

Grafica de m(i) vs m(i+1)

```
m_i = resultados$dx[-length(resultados$m_c)]
m_i2 = resultados$dx
m_i2 = m_i2[-1]

plot(x=m_i, y = m_i2, xlab = "error_i", ylab = "error_(i+1)",
     main = "Convergenca")
```

Convergencia



De acuerdo con la grafica el metodo tiene una convergencia lineal.

Caso especial: dos raices en el intervalo

```
resultados <- biseccion(f, 1, 30, 1e-7)

cat("Cero de f en [0,20] es approx: ",resultados$resultado, "con error <=",
    resultados$dx[resultados$i],"\n")
```

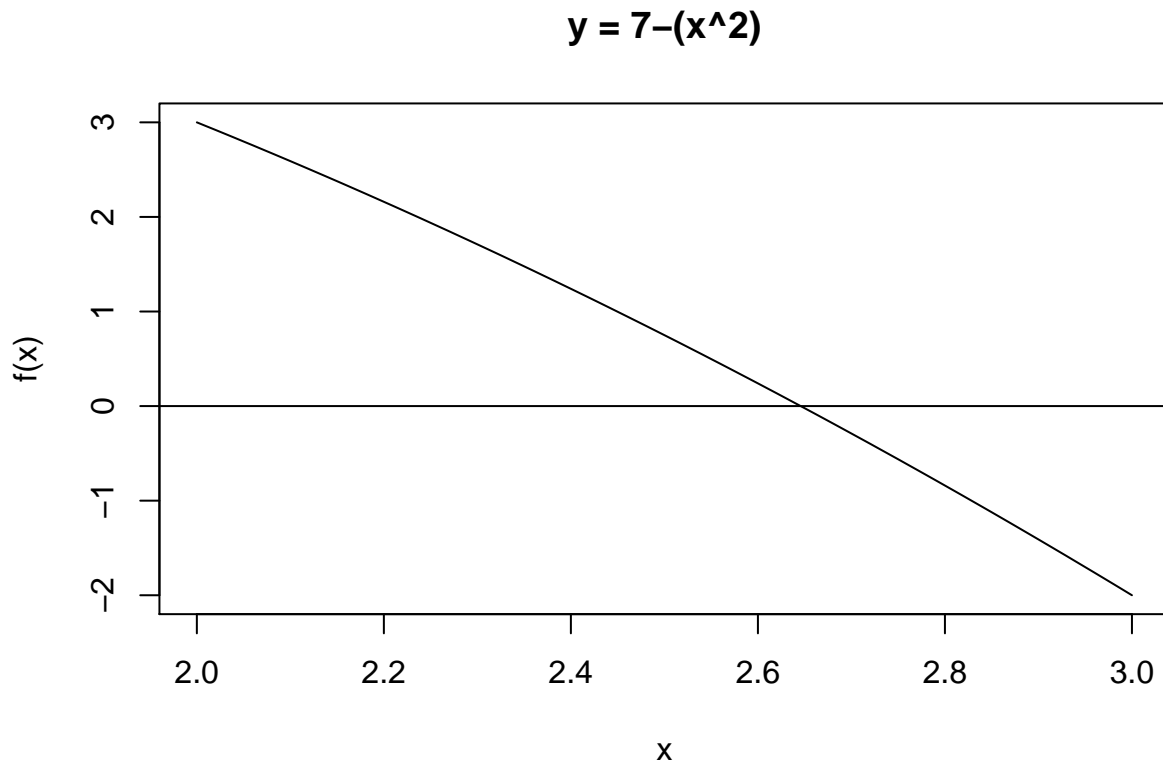
```
## Cero de f en [0,20] es approx: 18.2105 con error <= 5.401671e-08
```

El metodo de biseccion se acerca solamente a un de las dos raices.

Caso especial: sacar raiz de un numero

Gráfica de la función

```
f = function(x) 7-x^2
curve(f, 2,3); abline(h=0, v=0) #gráfico para decidir un intervalo
title(main="y = 7-(x^2)")
```



La raíz de 7 está en el intervalo $[2,3]$

Resultados

```
resultados <- biseccion(f, 2, 3, 1e-7)
cat("Cero de f en [2,3] es approx: ", resultados$resultado, "con error <=",
    resultados$dx[resultados$i], "\n")
```

```
## Cero de f en [2,3] es approx: 2.645751 con error <= 5.960464e-08
```

Cambiando los intervalos

```
a = c(2:0)
b = c(3:5)
iter = c()
error = c()
n = 1

for(num in b){
  iter = c(iter, biseccion(f, a[n], b[n], 1e-7)$i)
  error = c(error, biseccion(f, a[n], b[n], 1e-7)$dx[iter[n]])
  n = n+1
}

tabla <- data.frame(
  "a" = a,
```

```

    "b"= b,
    "b-a"= b-a,
    "iteraciones" = iter,
    "Error est." = error
)

```

```

print(tabla)

```

```

##   a b b.a iteraciones  Error.est.
## 1 2 3   1          23 5.960464e-08
## 2 1 4   3          24 8.940697e-08
## 3 0 5   5          25 7.450581e-08

```

Entre mas mas alejado estén los valores iniciales de a y b del valor de la raiz, mas iteraciones va a necesitar el metodo para llegar a la tolerancia dada.

Numero de iteraciones esperada

La formula para el numero minimo de iteraciones necesarias para tener un error menor a s es:

$$Iteraciones \geq \ln((b - a)/s)/\ln 2$$

Comprobacion:

```

cat("Numero minimo de estimaciones esperada para sacar la raiz de el interavalo [2,3]:"
    , log((3-2)/1e-7)/log(2))

```

```

## Numero minimo de estimaciones esperada para sacar la raiz de el interavalo [2,3]: 23.2535

```

Comparando con la tabla anterior se puede comprobar que numero de iteraciones necesarias fue 23.