Método de posición falsa

Juan Anagrita, Hector Hernandez, Aldemar Ramirez

Agosto 8,2019

Problema

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

Solución

Lenguaje de programación: R

Función principal-método de posición falsa

Parametros:

-f <- función

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

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

-errorMax <- tolerancia minima que debe tener la funcion

-iterMax <- iteraciones máximas que puede tener el método.

Valores de retorno:

-a_i <- valor de a traves de las iteraciones

-b_i <- valor de b traves de las iteraciones

-x_i <- valor de x traves de las iteraciones

-i <- iteraciones necesarias para llegar a la tolerancia

-errorA <- Error estimado absoluto.

-ErrorR <- Error estimado relativo

-resFinal <- valor final de x.

```
posFalsa = function(f,a,b,errorMax,iterMax = 100){

  if( sign(f(a)) == sign(f(b)) ){ stop("f(xa) y f(xb) tienen el mismo signo") }

  k = 0
  a_n = a;
  b_n = b;

  a_i = c()
  b_i = c()
  errorAbsoluto = c()
  errorRelativo = c()
  x_i = c()
```

```r
  e1_i = c()
  e2_i = c()

  x_n = 0

  repeat{

    x_anter = x_n
    x_n = (a_n*f(b_n)-b_n*f(a_n))/(f(b_n)-f(a_n))


    e1 = x_n-a_n
    e2 = b_n-x_n

    e1_i = c(e1_i,e1)
    e2_i = c(e2_i,e2)



    if(f(x_n)*f(a_n)>0){
      a_n = x_n
    }else{
      b_n = x_n
    }

    a_i = c(a_i,a_n)
    b_i = c(b_i,b_n)
    errorAbsoluto = c(errorAbsoluto,abs(x_anter-x_n))
    errorRelativo = c(errorRelativo,abs(x_anter-x_n)/abs(x_n))
    x_i = c(x_i,x_n)

    k = k+1

    if( f(x_n) == 0 ){
      res = list("iteraciones" = k, "error" = errorAbsoluto, "a_i"=a_i,
                 "errorR" = errorRelativo, "b_i" = b_i, "x_i" = x_i,
                 "resFinal" = x_n, "e_1" = e1_i, "e_2" = e2_i)
      return(res)
    }

    if(abs(x_anter-x_n)<=errorMax || k>=iterMax){
      res = list("iteraciones" = k, "errorA" = errorAbsoluto,
                 "errorR" = errorRelativo, "a_i"=a_i, "b_i" = b_i, "x_i" = x_i,
                 "resFinal" = x_n, "e_1" = e1_i, "e_2" = e2_i)
      return(res)
    }
  }
}
```

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

Formula general para calcular x_n

$$x_n = (a_n * f(b_n) - b_n * f(a_n))/(f(b_n) - f(a_n))$$

Dependiendo del valor de $f(x_n) * f(a_n)$ el algoritmo cambia el valor de a_n y b_n

si $f(x_n) * f(a_n) > 0$ entonces

$$a_n = x_n$$

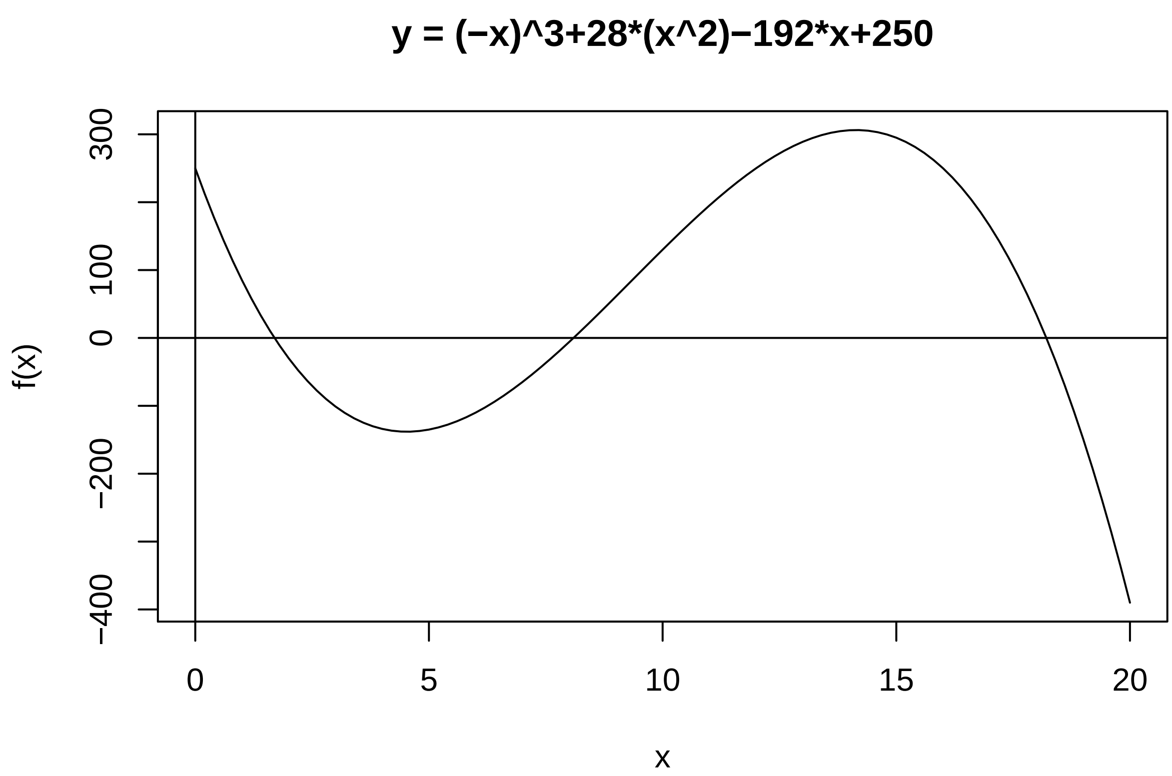
en caso contrario

$$b_n = x_n$$

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" )
```



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

Resultados de las raices

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

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

```
## Cero de f en [0,5] es approx:  1.696277 con error <= 3.246127e-08
```

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

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

```
## Cero de f en [5,10] es approx:  8.09322 con error <= 1.196755e-08
```

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

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

```
## Cero de f en [15,20] es approx:  18.2105 con error <= 2.429781e-08
```

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

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

```r
tabla <- data.frame(
  "iteraciones" = 1:resultados$iteraciones,
  "a" = resultados$a_i,
  "b" = resultados$b_i,
  "x" = resultados$x_i,
  "e1" = resultados$e_1,
  "e2" = resultados$e_2,
  "Error est." = resultados$errorA,
  "Error Rel." =  resultados$errorR
)
print(tabla)
```

```
##    iteraciones        a  b        x           e1       e2   Error.est.
## 1            1 17.15328 20 17.15328 2.153285e+00 2.846715 1.715328e+01
## 2            2 17.93661 20 17.93661 7.833204e-01 2.063395 7.833204e-01
## 3            3 18.14488 20 18.14488 2.082747e-01 1.855120 2.082747e-01
## 4            4 18.19509 20 18.19509 5.021331e-02 1.804907 5.021331e-02
## 5            5 18.20690 20 18.20690 1.180898e-02 1.793098 1.180898e-02
## 6            6 18.20966 20 18.20966 2.760804e-03 1.790337 2.760804e-03
## 7            7 18.21031 20 18.21031 6.445488e-04 1.789693 6.445488e-04
## 8            8 18.21046 20 18.21046 1.504302e-04 1.789542 1.504302e-04
## 9            9 18.21049 20 18.21049 3.510603e-05 1.789507 3.510603e-05
## 10          10 18.21050 20 18.21050 8.192577e-06 1.789499 8.192577e-06
## 11          11 18.21050 20 18.21050 1.911867e-06 1.789497 1.911867e-06
## 12          12 18.21050 20 18.21050 4.461637e-07 1.789497 4.461637e-07
## 13          13 18.21050 20 18.21050 1.041192e-07 1.789496 1.041192e-07
## 14          14 18.21050 20 18.21050 2.429781e-08 1.789496 2.429781e-08
##      Error.Rel.
## 1  1.000000e+00
## 2  4.367161e-02
## 3  1.147843e-02
```

```
## 4  2.759717e-03
## 5  6.485992e-04
## 6  1.516120e-04
## 7  3.539472e-05
## 8  8.260652e-06
## 9  1.927791e-06
## 10 4.498820e-07
## 11 1.049870e-07
## 12 2.450035e-08
## 13 5.717534e-09
## 14 1.334275e-09
```

Grafica de iteraciones vs error estimado

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

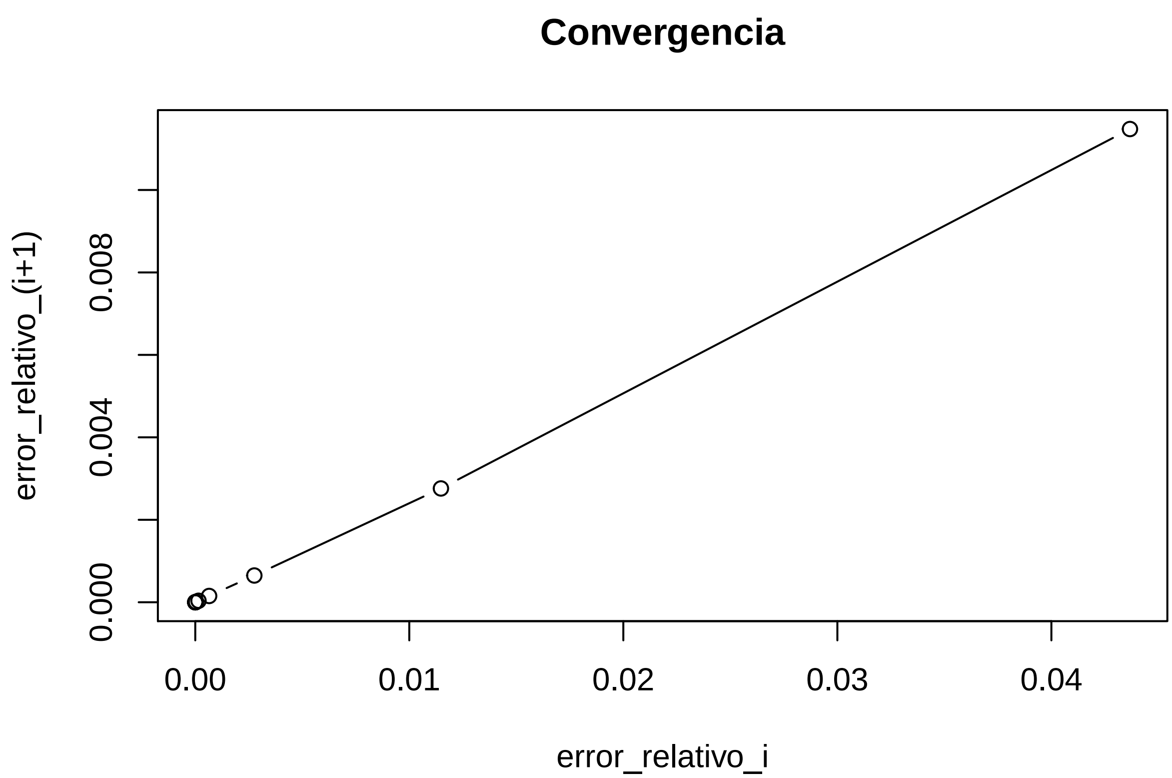
El error estimado siempre converge a cero.

Graficas de convergencia

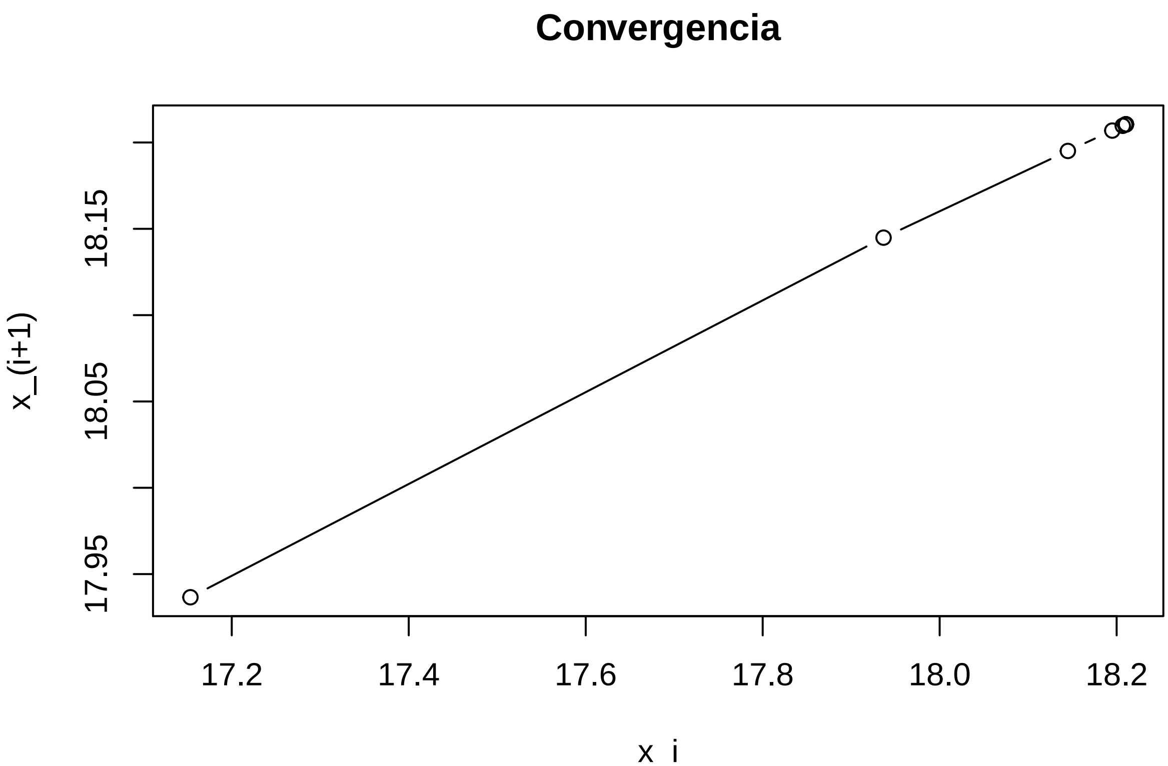
```
m_i = resultados$errorR[-length(resultados$a_i)]
m_i2 = resultados$errorR
m_i2 = m_i2[-1]
```

```r
m_i = m_i[-1]
m_i2 = m_i2[-1]


plot(x =m_i, y = m_i2, xlab = "error_relativo_i", ylab = "error_relativo_(i+1)",
     main = "Convergencia", type = "b")
```




```r
m_i = resultados$x_i[-length(resultados$a_i)]
m_i2 = resultados$x_i
m_i2 = m_i2[-1]
plot(x =m_i, y = m_i2, xlab = "x_i", ylab = "x_(i+1)",
     main = "Convergencia", type = "b")
```



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

Caso especial: dos raices en el intervalo

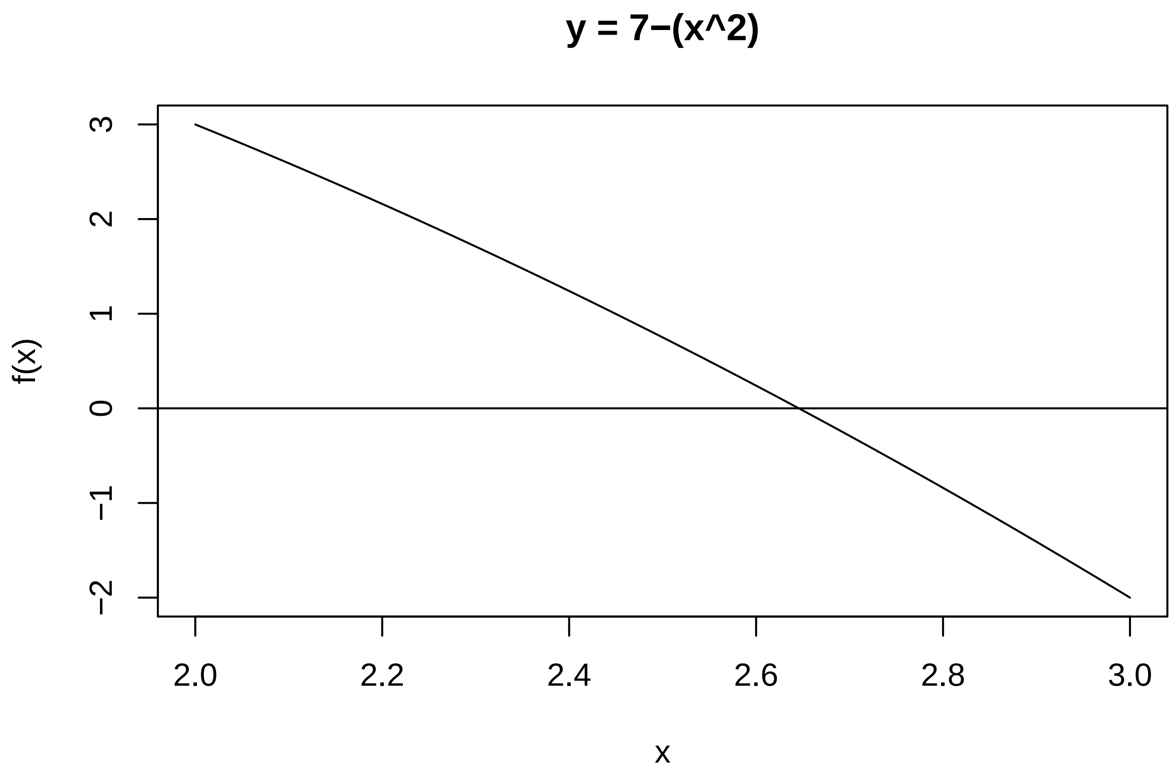
```
resultados <- posFalsa(f, 0, 20, 1e-7)
cat("Cero de f en [0,20] es approx: ",resultados$resFinal, "con error <=",
    resultados$errorA[resultados$iteraciones],"\n")
```

```
## Cero de f en [0,20] es approx:  1.696277 con error <= 7.071624e-08
```

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

Cambiando los intervalos [a,b]

```
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)")
```




```
resultados <- posFalsa(f, 2, 3, 1e-7)
cat("Cero de f en [2,3] es approx: ",resultados$resFinal, "con error <=",
    resultados$errorA[resultados$iteraciones],"\n")
```

```
## Cero de f en [2,3] es approx:  2.645751 con error <= 4.206924e-08
```

```
a = c(2:0)
b = c(3:5)
iter = c()
error =c()
n = 1
for(num in b){
  iter = c(iter,posFalsa(f, a[n], b[n], 1e-7)$iteraciones)
  error = c(error,posFalsa(f, a[n], b[n], 1e-7)$errorA[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           7 4.206924e-08
## 2 1 4   3          12 4.785076e-08
## 3 0 5   5          16 7.766411e-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 toleracia dada.