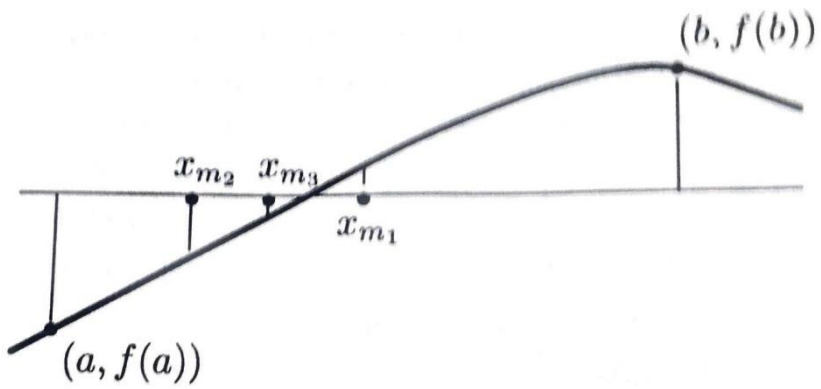
MÉTODO DE LA BISECCIÓN

El método de la bisección, es otro método cerrado, donde es necesario el ingreso de dos puntos iniciales de los cuales el programa partirá.

Este método consiste en que dados dos puntos iniciales $X_{inferior}$ y $X_{superior}$ o bien [a, b], el cual debe tener una raíz en él, el programa divide este intervalo en dos sub-intervalos de igual tamaño, ahora con los signos de los valores que corresponden a la evaluación de la función en los extremos de este intervalo y en el punto medo X_{medio}. A partir del teorema 2 (Existencia de raíces), se decide cuál de los intervalos contiene la raíz, y el programa se queda con ese. Aplicando estos pasos repetidamente, se logra una sucesión y se encuentran los valores siguientes a estos obteniendo los puntos medios de cada intervalo y aproximándose cada vez más a la raíz en cada iteración.

Una demostración gráfica de lo explicado anteriormente:



FIGURA

En la gráfica se observa el intervalo $\big(a, f(a)\big)\ y\ (b, f(b))$ donde su punto medio es x_{m1}, a continuación se verifica cuál de las dos partes divididas contiene una raíz, verificándolo mediante un cambio de signo. Ahora el intervalo pasa a ser $\big(a, f(a)\big)\ y\ x_{m1}$ y su punto medio x_{m2}, y así sucesivamente hasta que converge a la raíz de la función.

Para comprender mejor el efecto del método de la bisección, a continuación se presentara una forma tabular donde se muestran los valores que va arrojando cada iteración, respecto a los extremos de los intervalos $x_{inferior}$ y $x_{superior}$, el valor de la raíz en su punto medio x_m, el valor de esa raíz evaluada en la función $f\ (x_m)$ y el error, el cual sería un criterio de parada.

i	x_{inf}	x_{sup}	x_{m_i}	$f(x_{m_i})$	$Error$
1	2	3	2.5	-1.37	
2	2	2.5	2.25	$+9.52 * 10^{-1}$	$2.50 * 10^{-1}$
3	2.25	2.5	2.375	$-5.10 * 10^{-2}$	$1.25 * 10^{-1}$
4	2.25	2.375	2.3125	$+4.94 * 10^{-1}$	$6.25 * 10^{-2}$
5	2.3125	2.375	2.34375	$+2.32 * 10^{-1}$	$3.13 * 10^{-2}$
6	2.34375	2.375	2.359375	$+9.30 * 10^{-2}$	$1.56 * 10^{-2}$
7	2.359375	2.375	2.3671875	$+2.16 * 10^{-2}$	$7.81 * 10^{-3}$
8	2.3671875	2.375	2.37109375	$+1.45 * 10^{-2}$	$3.91 * 10^{-3}$
9	2.3671875	2.37109375	2.369140625	$+3.60 * 10^{-3}$	$1.95 * 10^{-3}$
10	2.369140625	2.37109375	2.370117188	$-5.45 * 10^{-3}$	$9.77 * 10^{-4}$
11	2.369140625	2.370117188	2.369628906	$-9.21 * 10^{-4}$	$4.88 * 10^{-4}$

TABLA

El ejemplo anterior se realizó con los siguientes datos:

$e^{3x-12} + xcos(3x) - x^2 + 4 = 0$

$Intervalo = [2,3]$

$Errorabsoluto\ (tolerancia) = 0.5\ x\ 10^{-3}$

Uno de los problemas que presenta este método, es que si se le ingresa una función que contiene más de una raíz, el programa no las encontrara todas ya que se limita a trabajar en un intervalo dado inicialmente, y aunque en ese intervalo hayan más de dos raíces, el programa convergerá solo a una de ellas.

PSEUDOCÓDIGO

Leer a, b, n, delta, tole

Contador = 0

Valorfuncion = delta + 1

Error = tole + 1

q = a

Mientras contador < n & valorfuncion > delta & error > tole **hacer**

contador = contador + 1

tabla (contador, 1) = contador

x = (a+b)/2

tabla (contador, 2) = x

y = función (x)

tabla (contador, 3) = y

valorfunción = absoluto (y)

error = absoluto (x-q)

tabla (contador, 4) = error

Sí función (a) * función (x) < 0 **entonces**

b = x

Sino

a = x

Fin si

q = x

Fin mientras

El comando tabla imprime los resultados en una tabla i, j.

CÓDIGO

```
function [ tabla ] = biseccion (a,b,n,delta,tole)


format long


i=0;
funcionvalue=delta+1;
error=tole+1;
q=a;
while i<n & funcionvalue>delta & error>tole
    i=i+1;
    tabla(i,1)=i;
    x=(a+b)/2;
    tabla(i,2)=x;
    y=funcion(x);
    tabla(i,3)=y;
    funcionvalue=abs(y);
    error=abs(x-q);% error absoluto
            %error=abs(error/x);% error relativo
    tabla(i,4)=error;
    if  funcion(a)*funcion(x)<0


       b=x;
      else
       a=x;
     endif
    q=x;
endwhile
disp('')
endfunction
```

EJEMPLO

Dada la ecuación $e^{3x-12} + xcos(3x) - x^2 + 4 = 0$, determinar una raíz de la ecuación con un máximo error absoluto de $0.5\ x\ 10^{-3}$.

```
Columns 1 through 3:

 1.000000000000000e+000  2.500000000000000e+000  -1.372302708874l9e+000
 2.000000000000000e+000  2.250000000000000e+000  9.520117939496O4e-001
 3.000000000000000e+000  2.375000000000000e+000  -5.09776367982520e-002
 4.000000000000000e+000  2.312500000000000e+000  4.935617474715l7e-001
 5.000000000000000e+000  2.343750000000000e+000  2.317721719231l2e-001
 6.000000000000000e+000  2.359375000000000e+000  9.297100773793l0e-002
 7.000000000000000e+000  2.367187500000000e+000  2.163364227487ll-002
 8.000000000000000e+000  2.371093750000000e+000  -1.451361082631l9e-002
 9.000000000000000e+000  2.369140625000000e+000  3.599720841650l2e-003
 1.000000000000000e+001  2.370117187500000e+000  -5.447032157182O1e-003
 1.100000000000000e+001  2.369628906250000e+000  -9.211757609879l6e-004

Column 4:

 5.000000000000000e-001
 2.500000000000000e-001
 1.250000000000000e-001
 6.250000000000000e-002
 3.125000000000000e-002
 1.562500000000000e-002
 7.812500000000000e-003
 3.906250000000000e-003
 1.953125000000000e-003
 9.765625000000000e-004
 4.882812500000000e-004
```

- En la primera columna se muestran el número de iteraciones realizadas.
- En la segunda columna se muestra el valor medio de cada intervalo.
- En la tercera columna se imprimen los valores del punto medio evaluado en la función.
- En la cuarta columna, se muestra el valor del error absoluto.

Como se puede observar, el programa llegó a la raíz en 11 iteraciones, con un error de $4.88\ x\ 10^{-4}$, lo cual indica que convergió antes de llegar a la tolerancia requerida que era de $0.5\ x\ 10^{-3}$.