

## **Laboratorio de métodos numéricos**

**Materia:** Métodos numéricos

**Alumno:** Cazajous Miguel

**Matricula:** 34980294

**Carrera:** Ingeniería en computación

**Fecha:** 03/05/11

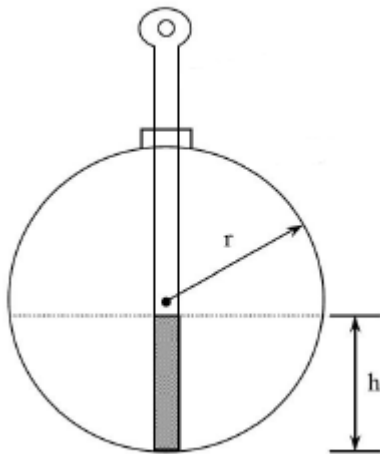
**E-mail:** migue\_143@hotmail.com

**Docente de laboratorio:** Ing. Gabriel Maggio

**Docente de teórico:** Beatriz Pedrotti

### **PROBLEMA PLANTEADO:**

-Problema completo adjunto al final del trabajo.



En principio, se deseaba desarrollar una forma de conocer el volumen de líquido que tenía almacenado un recipiente esférico de 2 metros de diámetro.

Para esto se procedió a calcularlo mediante una fórmula (1) conociendo su altura "h". La altura se medía observando donde quedaba la marca en una regla.

1-

$$V = \frac{\pi h^2 (3r - h)}{3}$$

Con "r" el radio interno del recipiente.

Luego se realizó la siguiente modificación al planteo anterior:

Ahora se busca leer directamente el volumen mediante una determinada longitud "h" que marca la regla.

Es este caso la resolución debe ser en forma inversa, es decir se debe determinar la altura a la que corresponde un determinado volumen de líquido contenido dentro del recipiente.

### **MODELO MATEMATICO:**

Veamos el modelo matemático empleado comenzando con ejemplo.

Como se mencionó anteriormente ahora se desea conocer la altura correspondiente a un volumen determinado.

EJ: suponemos que:

$$V = 1m^3$$

Y observando que el radio r es:

$$r = \frac{diam}{2} = \frac{2m}{2} = 1m$$

La ecuación 1 queda de la siguiente forma.

$$1 = \frac{\pi h^2(3x1 - h)}{3}$$

$$3 = \pi h^2(3 - h)$$

$$3 = 3\pi h^2 - \pi h^3$$

Ahora dividimos ambos miembros por  $-\pi$ :

$$-\frac{3}{\pi} = -3h^2 + h^3$$

Reordenando términos se tiene que:

$$h^3 - 3h^2 + \frac{3}{\pi} = 0$$

De forma más genérica es decir con un volumen "V" y un radio "r" nos queda:

$$h^3 - 3rh^2 + \frac{3V}{\pi} = 0$$

Donde la función a la que se le desea encontrar los ceros es:

$$f(h) = h^3 - 3rh^2 + \frac{3V}{\pi}$$

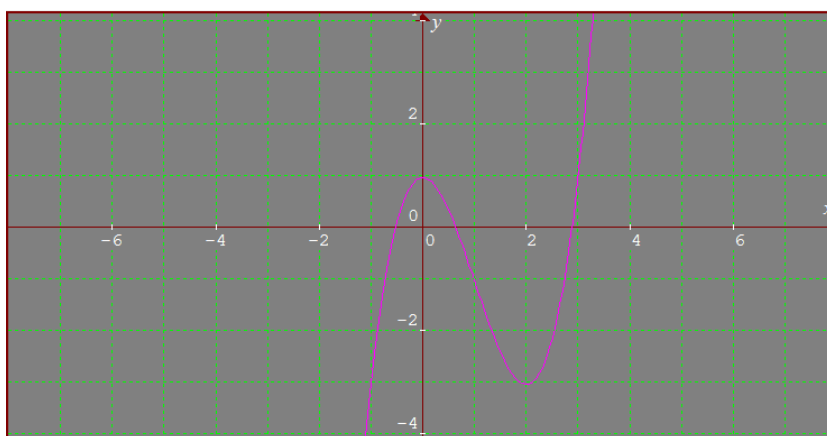
Lo que se deduce que altura "h" es la raíz positiva y menor que el diámetro del recipiente.

EJ:

Utilizando un programa para graficar funciones (FIG 1) podemos ver claramente que la raíz que nos interesa es única y es la raíz positiva y menor que el diámetro.

Para un volumen ingresado de 1 metro cubico se tiene la siguiente grafica en función de la altura (h)

FIG 1



### **CRITERIOS DE SELECCIÓN DEL METODO NUMERICO:**

Una vez conocida la función a la cual se le desea encontrar sus raíces (ceros), el siguiente paso es seleccionar el método numérico más apropiado para esta labor.

El método numérico seleccionado para realizar el cálculo de la raíz es el método de bisección.

Un pequeño resume de en qué consiste este método:

Primero este método es cerrado, ya que la raíz que se desea encontrar debe ser “encerrada” por dos valores iniciales.

Este método se basa en el teorema del valor medio, el cual enuncia que toda función continua en un intervalo  $[a, b]$  toma todos valores que se encuentran entre  $f(a)$  y  $f(b)$ . Sea que un valor entre  $f(a)$  y  $f(b)$  es la imagen de por lo menos un valor  $x \in \text{dom}f \cap [a, b]$ , es decir pertenece al dominio de  $f$  y a su vez esta entre  $a$  y  $b$ .

Entonces si  $f(a)$  y  $f(b)$  tienen signos opuesto existe por lo menos un valor  $p \in [a, b]$ , tal que  $f(p)=0$ .

EL objetivo de este método entonces es encontrar un valor entre  $[a,b]$  para el cual la función sea nula.

Una condición necesaria para el funcionamiento de este método (y todos los cerrados) es que el intervalo inicial  $[a, b]$  encierre la raíz, puesto que de lo contrario no existiría cambio de signo en  $f(a)$  y  $f(b)$

El método opera de la siguiente forma:

Se divide el intervalo  $[a, b]$  por la mitad, luego se verifica que mitad posee el cambio de signo, por consiguiente ese intervalo incluirá la raíz.

Para observar donde se produce el cambio de signo simplemente se realiza el producto de la función valuada en esos puntos, suponemos que  $c$  es el valor intermedio entre  $a$  y  $b$  entonces sí:

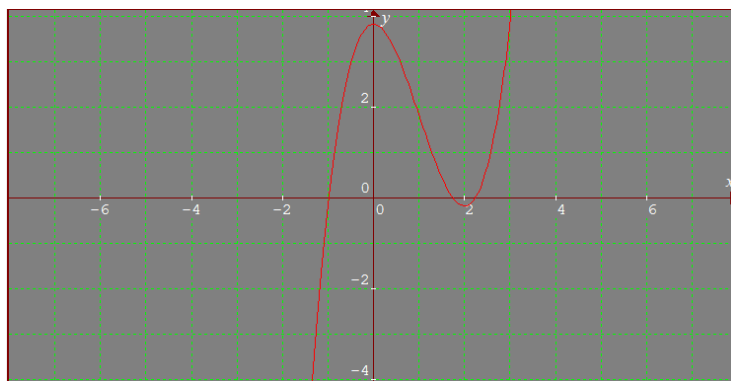
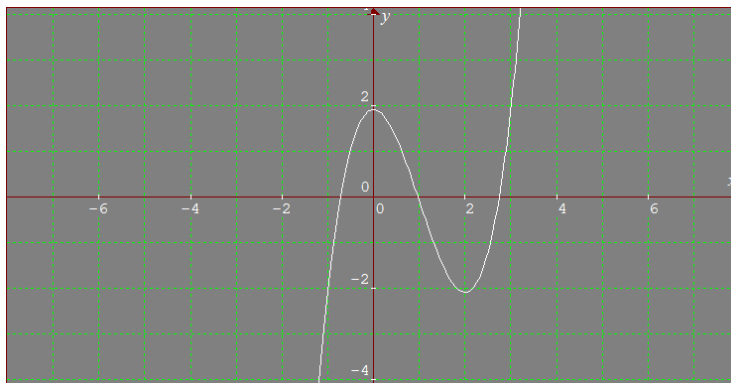
$$f(a) \times f(c) < 0$$

la raíz está en la mitad del intervalo inicial, por lo que ahora nuestro intervalo es  $[a, c]$ . entonces el intervalo se va reduciendo cada vez mas hasta converger al valor  $x$  para el cual la función se anula (raíz)

Veamos algunas de sus ventajas para este trabajo en particular.

- Principalmente el método debía pertenecer a los cerrados, ya que son métodos que convergen siempre que se elija correctamente el intervalo inicial, y como hemos visto en este problema eso es muy sencillo.  
Sabemos que el intervalo no debe pasar los límites  $[0, \text{diam}]$ , puesto que no existe valor negativo para la altura y esta no puede ser mayor que el diámetro del recipiente.
- Ahora la pregunta sería por qué no otro método cerrado. Un método cerrado que agiliza la convergencia hacia la raíz respecto del método de bisección es el método de falsa posición, el cual no divide el intervalo a la mitad sino que traza una recta entre los valores  $f(a)$  y  $f(b)$  lo que hace que el intervalo no sea dividido a la mitad. Pero este método tiene un problema, si la función es muy pronunciada cerca de la solución se estanca y converge lentamente.

Algunas imágenes como ejemplo. Para  $r=1$  y  $\text{vol} = 2$  y  $\text{vol} = 4$



La solución para esto es el método de falsa posición modificado.

- Por esto entonces se escogió el método de bisección, el cual posee un algoritmo muy sencillo en comparación con el método de falsa posición modificado. Además realizando el cálculo de iteraciones necesarias. Son números no muy grandes lo que proporciona una solución eficaz, rápida, y sencilla.

Fórmula para el cálculo de iteraciones

$$n > \frac{\ln \left( \frac{b-a}{\delta} \right)}{\ln(2)} > K$$

Veamos un ejemplo. Suponemos que el intervalo es [0, 5] y queremos una tolerancia  $\delta = 0.001$

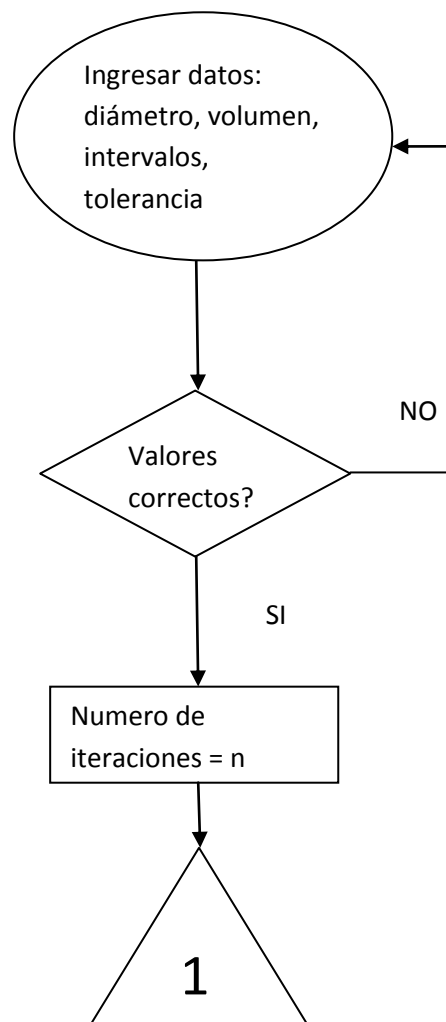
Entonces:

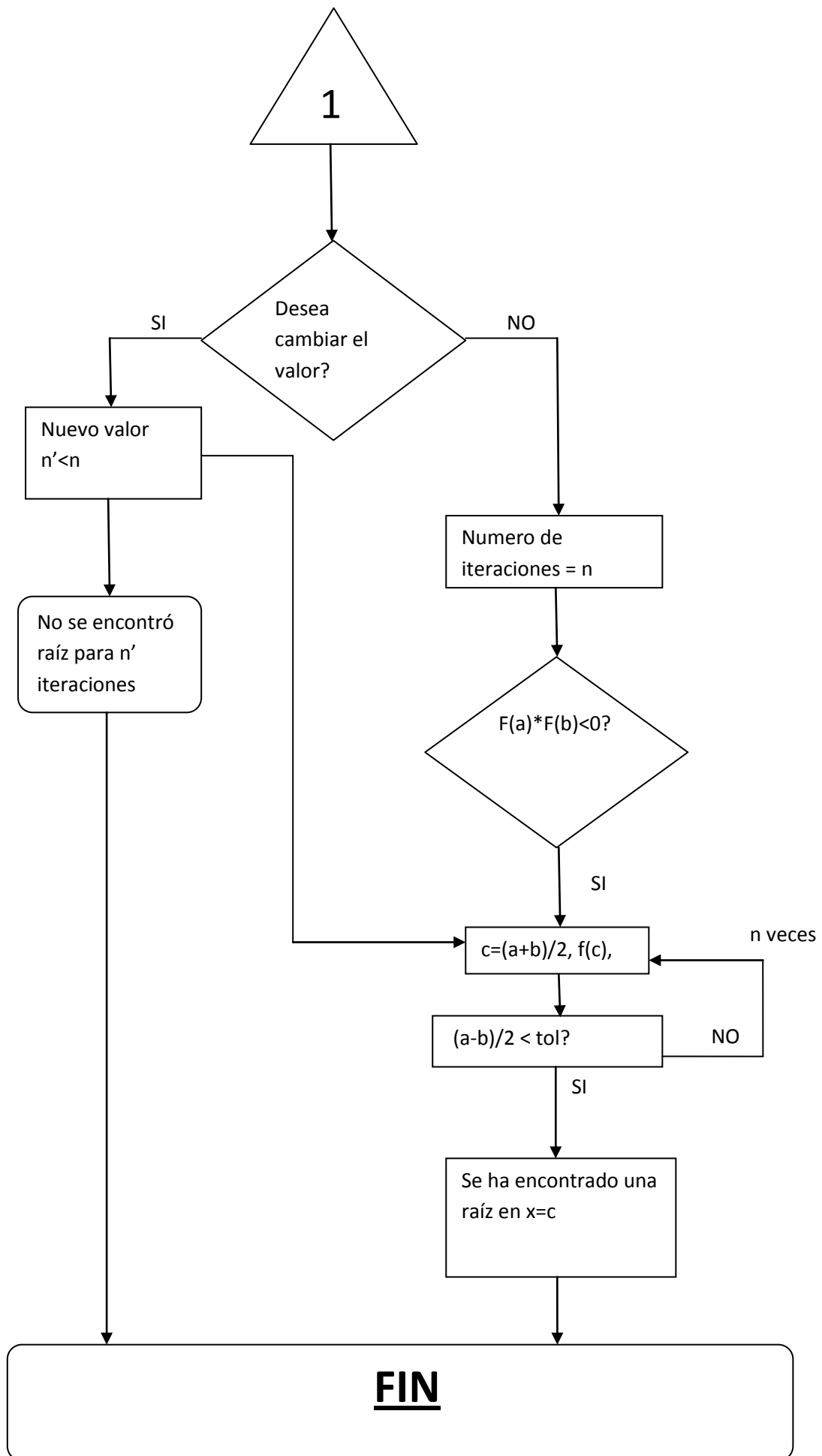
$$n > \frac{\ln \left( \frac{5}{0.001} \right)}{\ln(2)} > 12.28$$

Lo cual lleva a que la solución se encuentra en la iteración  $n=13$ . Que no es un valor demasiado grande.

- Además este método a diferencia de el método de falsa posición, nos permite saber de antemano cuantas iteraciones serán necesarias para encontrar la raíz, y como hemos observado no son demasiadas las iteraciones para encontrar un valor realmente preciso.

#### REPRESENTACION DEL ALGORITMO MEDIANTE UN DIAGRAMA DE FLUJO:





### **CODIGO FUENTE:**

```
#include<iostream>
#include<cmath>

using namespace std;

double funcion(double);
void biseccion (double, double, double, int);

double diam;
const double diammax = 40;
double vol;
double volmax;
const double pi = 3.1415926;
double a, b, c;
double tol;
double radio;
int itermax;
int iter;
double interin;
int opcion;

int main()
{
    cout << "Este programa determinara la altura de liquido dentro de un recipiente esferico" << endl;
    cout << "\nIngrese un diametro para el recipiente en metros " << endl;
    cout << "\n";
    cin >> diam;

    if (diam < 0 || diam > diammax)

        do
        {
            cout << "\nValor negativo o muy grande ingrese un valor positivo y menor o igual que " << diammax << endl;
            cin >> diam;
        }

        while (diam < 0 || diam > diammax);

    radio = diam/2;

    cout << "\nEl diametro de la esfera es " << diam << " metro/s" << endl;

    volmax = (4*pi/3)* pow(diam,3)/8;

    cout << "\nEl volumen maximo permitido es " << volmax << " metros cubicos" << endl;
    cout << "\nIngrese un valor de volumen en metros cubicos " << endl;
    cout << "\n";
```



```

cin >> vol;

if (vol < 0 || vol > volmax)

do
{
    cout << "\nValor negativo incorrecto o demasiado grande " << endl;
    cout << "\ningrese un volumen positivo menor que " << volmax << endl;
    cout << "\n";

    cin >> vol;
}

while(vol < 0 || vol > volmax);

cout << "\nEl volumen ingresado es " << vol << " metros cubicos" << endl;
cout << "\nLa altura (h) correspondiente a ese volumen es la raiz positiva y menor que " <<
diam << endl;
cout << "\nde la ecuacion: " << endl;
cout << "\nf(h)=h^3-3rh^2+3*" << vol << "/pi" << endl;
cout << "\nSiendo (r) el radio de la esfera " << endl;
cout << "\n";

system("pause");

cout << "\nIngrese valores para el intervalo que encierra la raiz " << endl;
cout << "\nEl limite inferior debe ser mayor o igual a 0 " << endl;
cout << "\ny el limite superior debe ser menor o igual que " << diam << endl;

do
{
    cout << "\nRecuerde que los valores deben encerrar la raiz " << endl;
    cout << "\n";
    cin >> a;
    cout << "\n";
    cin >> b;

if(a<0 || b<0 || a>diam || b>diam || a==b)

{
    do
    {

        cout << "\nLos valores ingresados son incorrectos, no ingrese valores negativos, " << endl;
        cout << "\niguales entre si y/o mayores a " << diam << endl;
        cout << "\nIngrese nuevamente los valores " << endl;
        cout << "\n";
        cin >> a;
        cout << "\n";
        cin >> b;
    }
}

```

```

    while (a<0 || b<0 || a>diam || b>diam || a==b);
}

if(a<b)

    cout << "\nEl intervalo ingresado es [" << a << ", " << b << "]" << endl;

    else
    {
        c=b;
        b=a;
        a=c;

        cout << "\nEl intervalo ingresado es [" << a << ", " << b << "]" << endl;

    }
}

while(funcion(a)*funcion(b)>0.0);

cout << "\nIngrese un valor de tolerancia (tol) " << endl;
cin >> tol;

if (tol<0)

    do
    {
        cout << "\nIngrese un numero positivo " << endl;
        cout << "\n";
        cin >> tol;
    }

    while(tol<0);

    cout << "\nLa tolerancia es " << tol << endl;

    interin = b-a;

    itermax = int((log(interin/tol)/log(2))+1);

    cout << "\nLa cantidad de iteraciones necesarias son " << itermax << endl;
    cout << "\nPresione (1) si desea restringir el numero de iteraciones " << endl;
    cout << "\n";
    cin >> opcion;

    if (opcion==1)
    {
        cout << "\nIngrese un nuevo valor de iteraciones " << endl;
        cout << "\n";
    }

```

```

do
{
    cout << "\nEl valor ingresado debe ser mayor a cero y menor o igual que " << itermax << endl;
    cout << "\n";
    cin >> iter;
}

while(iter <= 0 || iter > itermax);

itermax=iter;

cout << "\nEl programa finalizara al llegar a las " << itermax << " iteraciones" << endl;
}

else

cout << "\nEl programa finalizara al llegar a las " << itermax << " iteraciones" << endl;
cout << "\n";

system("pause");

biseccion(a, b, tol, itermax);

cout << "\n";

system("pause");

return 0;

}

void biseccion(double a, double b, double tol, int itermax)
{
    int i;

    double x1, x2, x3;
    double f1, f2, f3;

    double ancho_it;

    cout << "\nel intervalo inicial es " << interin << endl;

    x1 = a;
    x3 = b;
    f1 = funcion(x1);
    f3 = funcion(x3);

```

```

for(i=0; i < itermax; i++)
{
    x2 = (x1 + x3)/2.0;
    f2=funcion(x2);

    if(f1 * f2 < 0.0)
    {
        ancho_it =(x2-x1)/2.00;
        f3 = f2;
        x3 = x2;
    }

    else
    {
        ancho_it =(x3-x2)/2.00;
        f1 = f2;
        x1 = x2;
    }

    if(abs(ancho_it) < tol)
    {
        cout << "\nSe encontro una raiz en X= " << x2 << endl;
        cout << "\nSe realizaron "<< i << " iteraciones " << endl;
        cout << "\nLa funcion valuada en " << x2 << " es igual a " << f2 << endl;

        return;
    }
}

cout << "\nNo se encontro ninguna raiz en " << itermax << " iteraciones" << endl;
cout << "\nque cumplan con el criterio de convergencia" << endl;

return;

}

double funcion(double x)
{
    double f;

    f = pow(x,3)-3*radio*pow(x,2)+3*vol/pi;

    return f;
}

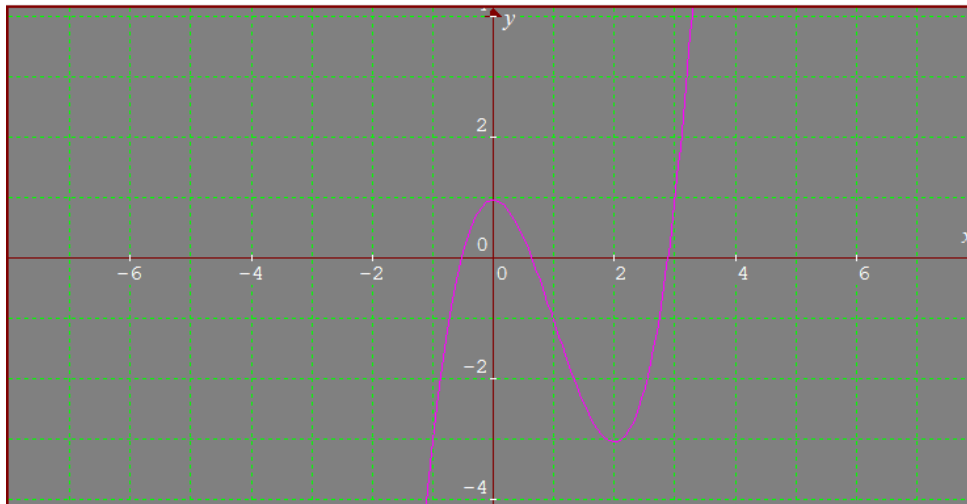
```

### **EVIDENCIA DE RESULTADOS DE EJECUCION:**

Propongamos resolver el ejemplo que muestra el problema, donde el diámetro es 2 metros, por consiguiente el radio es 1 metro, el volumen que se ingresa es 1 metro cubico.

Probemos primero con una tolerancia igual a 0.1 y luego con una de 0.0001 y veremos los resultados obtenidos.

Este es el resultado usando un programa para graficar funciones.



Podemos ver a simple vista que el valor de la raíz está entre 0 y 1, más precisamente entre 0.5 y 1.

La formula de la función es la siguiente:

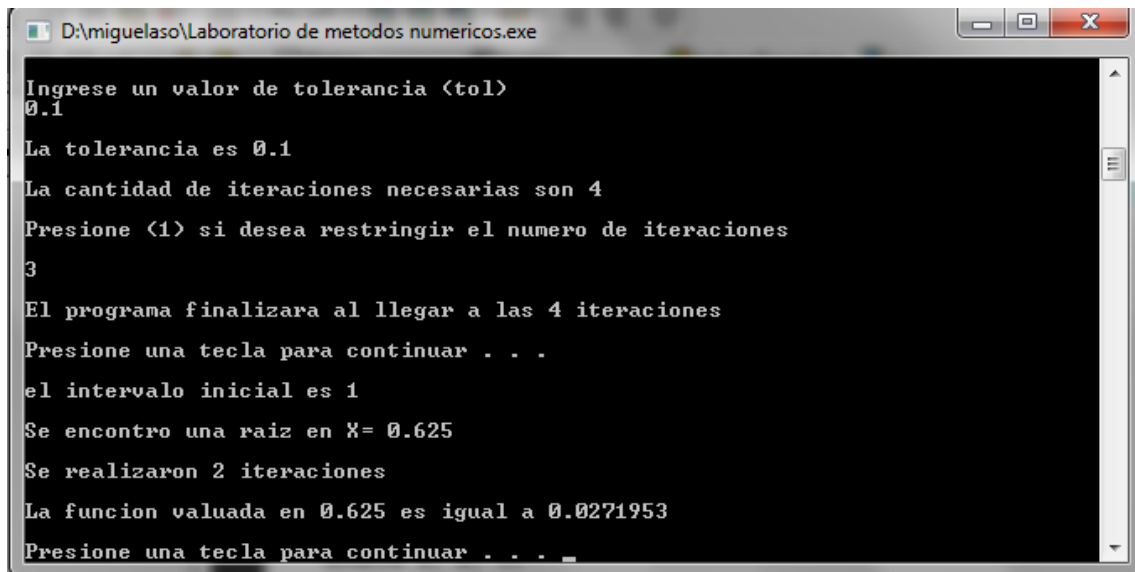
$$f(h) = h^3 - 3rh^2 + \frac{3V}{\pi}$$

Como  $r = 1$  y  $\text{vol} = 1$

$$f(h) = h^3 - 3h^2 + \frac{3}{\pi}$$

Veamos ahora el resultado obtenido mediante el programa con el método de bisección.

### Resultado para tolerancia igual a 0.1

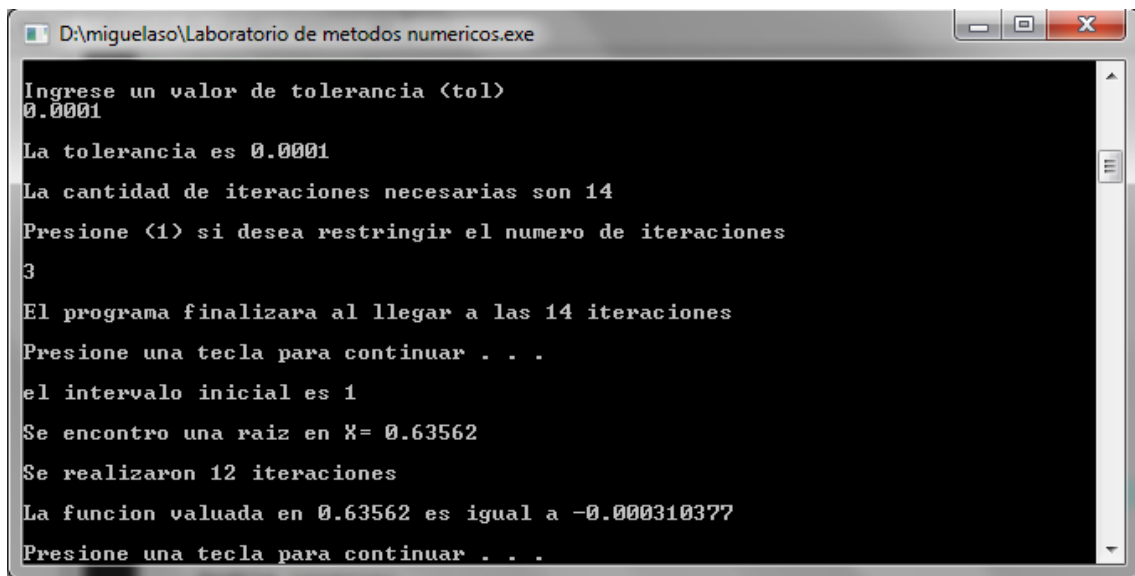


```
D:\miguelaso\Laboratorio de metodos numericos.exe

Ingrese un valor de tolerancia <tol>
0.1
La tolerancia es 0.1
La cantidad de iteraciones necesarias son 4
Presione <1> si desea restringir el numero de iteraciones
3
El programa finalizara al llegar a las 4 iteraciones
Presione una tecla para continuar . . .
el intervalo inicial es 1
Se encontro una raiz en X= 0.625
Se realizaron 2 iteraciones
La funcion valuada en 0.625 es igual a 0.0271953
Presione una tecla para continuar . . .
```

La raíz  $x = 0.625$  con 2 iteraciones y la función valuada en el punto es  $f(x)=0.0271953$

### Resultado para tolerancia igual a 0.0001



```
D:\miguelaso\Laboratorio de metodos numericos.exe

Ingrese un valor de tolerancia <tol>
0.0001
La tolerancia es 0.0001
La cantidad de iteraciones necesarias son 14
Presione <1> si desea restringir el numero de iteraciones
3
El programa finalizara al llegar a las 14 iteraciones
Presione una tecla para continuar . . .
el intervalo inicial es 1
Se encontro una raiz en X= 0.63562
Se realizaron 12 iteraciones
La funcion valuada en 0.63562 es igual a -0.000310377
Presione una tecla para continuar . . .
```

La raíz  $x = 0.63562$  con 12 iteraciones y la función valuada en el punto es  $f(x)=-0.000310377$

**CONCLUSION:**

Como se puede observar el valor es muy preciso, y cumple con lo que muestra el grafico. Al poner una tolerancia menor, es decir la distancia entre el valor de la raíz real y el calculado es más pequeño, obtenemos una aproximación muy confiable.

De esta forma demostramos la utilidad de los métodos numéricos, para resolver problemas en los cuales las soluciones analíticas y graficas no nos proporcionan un resultado muy exacto, en este caso el método de bisección, nos brinda una herramienta eficaz para determinar la raíz de la ecuación.

Además el implemento de estos métodos para resolver problemas reales, como el que se presento aquí, nos ayuda a mejorar nuestras técnicas de programación, haciendo que implementemos nuestras propias aplicaciones para resolverlos y para los cuales no existen programas comerciales que nos ayuden.