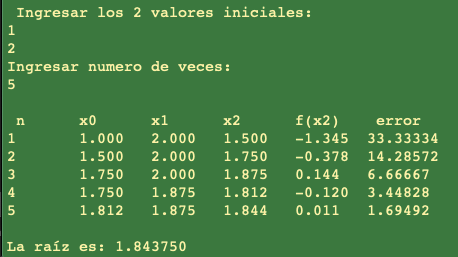
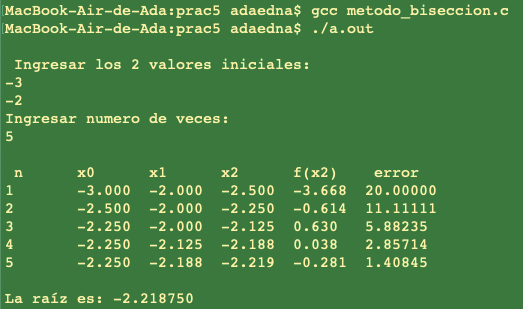
Práctica 5: Métodos cerrados, bisección y falsa posición

Samuel Von-Borsted, grupo 28, profesor Roilhi Frajo Ibarra Hernández 30 de noviembre de 2020

1. **Tomar como ejemplo las funciones** 2*x* cos(2*x*) (*x* + 1)2 **en un intervalo** [ 3*,* 2] **y** ln *x* + *x*2 4 **en un intervalo** [1*,* 2]

*− | | −*

*−*



(a) 2*x* cos(2*x*) *−* (*x* + 1)2 (b) ln*|x|* + *x*2 *−* 4

Figura 1: Tablas de las dos funciones

## 1.. \*\*\* IMPLEMENTACION DEL METODO DE BISEC-

**CION \*\*\***

#include <stdio.h> #include <math.h>

float ftrig(float x); float flog(float x);

/\*void biseccion(x0,x1,x2,n);

Programa para encontrar raices por el metodo de biseccion \*/

float ferrabs(float x, float y);

void biseccion(float x0, float x1,int n,float (\*f)(float), float (\*g)(float, float));

int main()

{

float x0, x1; int e, step = 1;

printf("\n Ingresar los 2 valores iniciales:\n"); scanf(" f f", &x0, &x1);

printf("Ingresar numero de veces:\n"); scanf(" d", &e);

/\* metodo de biseccion \*/ biseccion(x0,x1,e,ftrig,ferrabs);

/\* segunda funcion\*/

printf("\n Ingresar los 2 valores iniciales:\n"); scanf(" f f", &x0, &x1);

printf("Ingresar numero de veces:\n"); scanf(" d", &e);

biseccion(x0,x1,e,flog,ferrabs); return 0;

}

/\*

Ecuaciones de las funciones.

\*/

float ftrig(float x) {

return (2\*x\*cos(2\*x))-pow((x+1),2);

}

float flog(float x) { return log(x)+pow(x,2)-4;

}

float ferrabs(float x, float y){

return 100\*fabs((x-y)/y);}/\*funcion de error \*/

/\*metodo de biseccion, rengon 43\*/

void biseccion(float x0,float x1, int n, float (\*f)(float), float (\*g)(float,float))

{

float f0, f1, x2,f2, error; int vez =1;

/\* calculando el valor de las funciones \*/ f0 = (\*f)(x0);

f1 = (\*f)(x1);

/\* revisando si la raiz esta entre los limites. \*/

while( f0 \* f1 > 0.0)

{

printf("valores inadecuados.\n");

printf("\n Ingresar los 2 valores iniciales:\n"); scanf(" f f", &x0, &x1);

f0 = (\*f)(x0);

f1 = (\*f)(x1);

}

printf("\n n\tx0\tx1\tx2\tf(x2)\t error\n");

do

{

x2 = (x0 + x1)/2; f2 = (\*f)(x2);

error = (\*g)(x1,x2);

printf(" d\t 3.3f\t 3.3f\t 3.3f\t 3.3f\t 2.5f\n", vez, x0, x1, x2, f2,error);

if( f0 \* f2 < 0)

{

}

else

{

}

x1 = x2; f1 = f2;

x0 = x2; f0 = f2;

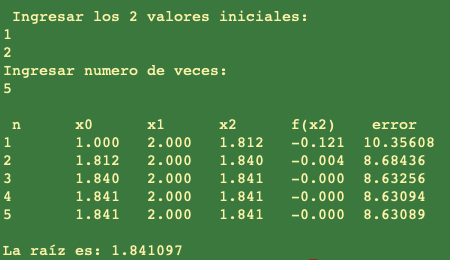
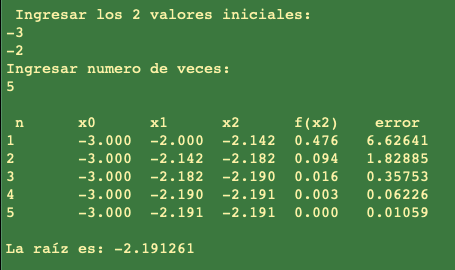
vez = vez + 1;

}while(vez<=n);

printf("\nLa raíz es: f\n", x2);

}

# Método de la falsa posición



(a) 2*x* cos(2*x*) *−* (*x* + 1)2 (b) ln*|x|* + *x*2 *−* 4

Figura 2: Tablas de las dos funciones

## 1.. IMPLEMENTACION DE LA FALSA POSICION

#include <stdio.h> #include <math.h>

float ftrig(float x); float flog(float x);

/\*void falsa\_posicion(x0,x1,x2,n);

Programa para encontrar raices por el metodo de falsa posicion \*/ float ferrabs(float x, float y);

void falsa\_posicion(float x0, float x1,int n,float (\*f)(float), float(\*g)(float, float)); /\*renglon 44\*/

int main()

{

float x0, x1; int e, vez = 1;

printf("\n Ingresar los 2 valores iniciales:\n"); scanf(" f f", &x0, &x1);

printf("Ingresar numero de veces:\n"); scanf(" d", &e);

/\* metodo de falsa posicion \*/ falsa\_posicion(x0,x1,e,ftrig,ferrabs);

/\* segunda funcion\*/

printf("\n Ingresar los 2 valores iniciales:\n"); scanf(" f f", &x0, &x1);

printf("Ingresar numero de veces:\n");

scanf(" d", &e);

falsa\_posicion(x0,x1,e,flog,ferrabs); return 0;

}

/\*

Ecuaciones de las funciones.

\*/

float ftrig(float x) {

return (2\*x\*cos(2\*x))-pow((x+1),2);

}

float flog(float x) { return log(x)+pow(x,2)-4;

}

/\*funcion de error\*/

float ferrabs(float x, float y){ return 100\*fabs((y-x)/y);}

/\* funcion de falsa posicion, renglon 44\*/

void falsa\_posicion(float x0,float x1, int n, float (\*f)(float), float(\*g)(float,float))

{

float f0, f1, x2,f2, error; int vez =1;

/\* calculando el valor de las funciones \*/ f0 = (\*f)(x0);

f1 = (\*f)(x1);

/\* revisando si la raiz esta entre los limites. \*/ while( f0 \* f1 > 0.0)

{

printf("valores inadecuados.\n");

printf("\n Ingresar los 2 valores iniciales:\n"); scanf(" f f", &x0, &x1);

f0 = (\*f)(x0);

f1 = (\*f)(x1);

}

printf("\n n\tx0\tx1\tx2\tf(x2)\t error\n");

/\* implementacion de falsa posicion \*/ do

{

x2 = x0 - (x0-x1) \* f0/(f0-f1); f2 = (\*f)(x2);

error = (\*g)(x1,x2);

printf(" d\t 3.3f\t 3.3f\t 3.3f\t 3.3f\t 2.5f\n",vez, x0, x1, x2, f2,error); if(f0\*f2 < 0)

{

}

else

{

}

x1 = x2; f1 = f2;

x0 = x2; f0 = f2;

vez = vez + 1;

}while(vez<=n);

printf("\nLa raíz es: f\n", x2);

}

# Resultados

Aunque el error en el método de la bisección es muy grande al inicio compa- rado con el método de la falsa posición, vemos que este último tiende a estan- carse en algunos casos, aunque las primeras iteraciones tengan un error menor a iteraciones mayores del método de la bisección.