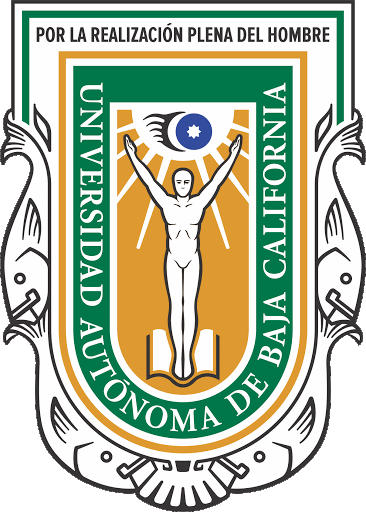
Ensenada, Baja California a 05 de noviembre del 2020



Practica No.5

Métodos cerrados: de la bisección y de la falsa posición.

Profesor: Roilhi Frajo Ibarra Hernández

Alumno: Fabian Diaz Fajardo

Grupo: 021

**Desarrollo de la practica.**

Tomar como ejemplo las funciones: f(x)=2 x cos(2 x)-(x+1)^(2) en un intervalo [−3, −2] y

f(x)=ln(abs(x))+x^(2)-4 en un intervalo [1, 2]

-Realizar un programa en lenguaje C donde se realice lo siguiente:

- Codificar las funciones, es decir, que sean definidas por el usuario (la trigonométrica podría

llamarse por ejemplo “funcion1” y la logarítmica “funcion2”).

- Codificar como funciones los métodos de bisección y de la falsa posición. Se debe recordar

que para el primer caso se necesita definir también la derivada de la función.

- Codificar una función de error absoluto.

**Función 1.**

**Error absoluto y función.**

#include <stdio.h>

#include <math.h>

float funcion1(float x\_entrada);

float ErrorRelativo(float ValorAnalitico,float ValorNumerico);

float ValorNumerico, ValorAnalitico, x\_ent ;

int main()

{

x\_ent=3;

ValorAnalitico= -10.2389;

ValorNumerico= -10.2390;

printf ("para la funcioon1 f(3) es igual a %2.4f\n", funcion1(x\_ent));

printf ("El Error relativo porcetual = %2.4f %%\n", ErrorRelativo(ValorAnalitico, ValorNumerico));

}

float funcion1(float x\_entrada){

// 2x\*cos(2x)-(x+1)^2x

float fx;

fx=(2\*(x\_entrada)\*cos(2\*x\_entrada))-pow((x\_entrada+1),2);

return fx;

}

float ErrorRelativo(float ValorAnalitico, float ValorNumerico){

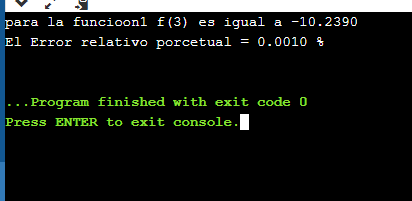
//epsilon= abs (ValorAnalitico-Valornumerico/ValorNumerico)\*100

float epsilon;

epsilon= fabs((ValorAnalitico - ValorNumerico)/ValorNumerico)\*100;

return epsilon;

}



**Método de bisección.**

#include <stdio.h>

#include <math.h>

#define RAIZ(x) (2\*(x)\*cos(2\*x))-pow((x+1),2)

int main() {

float a, b, xi, xa, Ea;

int iteraciones;

printf("Dame el valor de b: ");

scanf("%f", &b);

printf("Dame el valor de a: ");

scanf("%f", &a);

iteraciones = 0;

xa = 0.0;

Ea = 0;

printf("%12s %10s %10s %10s %10s\n","Iteraciones", "b", "a", "Xi","Error Aproximado");

do{

iteraciones++;

xi = (b + a) / 2;

if(iteraciones >= 0) {

Ea = ((xi - xa) / xi) \* 100; // Error aproximado

}

Ea = fabs(Ea); // valor absoluto del número

printf("%12d %10f %10f %10f %10f\n",iteraciones, b, a, xi, fabs(Ea));

xa = xi;

if(RAIZ(xi) > 0) {

b = xi;

} else{

a = xi;

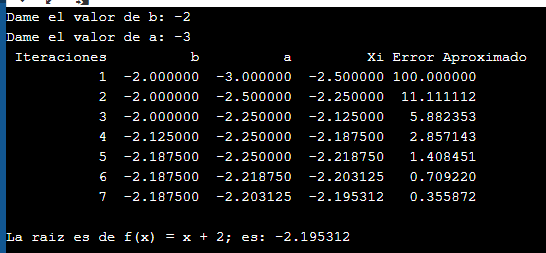
}

} while(Ea > 0.5 || (Ea == 0.0 && iteraciones == 1));

printf("\nLa raiz es de f(x) = x + 2; es: %f\n", xi);

return 0;

}



**Método de falsa posición.**

#include <stdio.h>

#include <math.h>

float

funcion1 (float x)

{

float y = 2 \* (x) \* cos (2 \* x) - pow ((x + 1), 2);

return y;

}

float

falsaposicion (float a, float b, int iter)

{

float xi, xi\_anterior = 0.0, errorRP;

int i = 1;

printf ("i \t a \t\t b \t\t xi \t\t Erp\n");

do

{

xi = b - ((funcion1 (b) \* (a - b)) / (funcion1 (a) - funcion1 (b)));

errorRP = ((xi - xi\_anterior) / xi) \* 100;

xi\_anterior = xi;

printf ("%d \t %.4f \t %.4f \t %.4f \t%.4f \n", i, a, b, xi,

fabs (errorRP));

if (funcion1 (a) \* funcion1 (xi) < 0)

{

b = xi;

}

else

{

a = xi;

}

i++;

}

while (funcion1 (a) \* funcion1 (xi) != 0 && i <= iter);

return xi;

}

int

main ()

{

float a = -3, b = -2, iteraciones = 5;

float raiz;

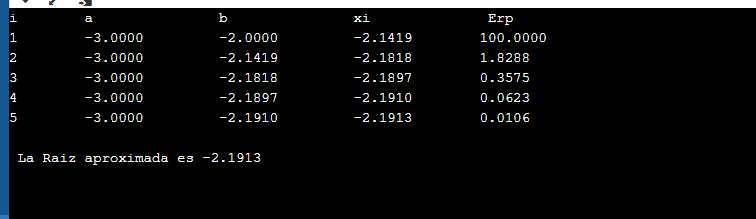
raiz = falsaposicion (a, b, iteraciones);

printf ("\n La Raiz aproximada es %.4f", raiz);

printf ("\n\n");

return 0;

}



**Función 2**

**Error absoluto y función.**

#include <stdio.h>

#include <math.h>

float funcion2(float x\_entrada);

float ErrorRelativo(float ValorAnalitico,float ValorNumerico);

float ValorNumerico, ValorAnalitico, x\_ent;

int main()

{

x\_ent= 5;

ValorAnalitico= 22.6094;

ValorNumerico= 22.6666; // Este numero es solo para saber si funciona el error relativo

printf("para la funcion2 f(5) es igual a %2.4f\n", funcion2(x\_ent));

printf ("El Error relativo porcetual = %2.4f %%\n", ErrorRelativo(ValorAnalitico, ValorNumerico));

}

float funcion2(float x\_entrada){

// In|x| + x^2 − 4

float fx;

fx= (log(x\_entrada)) + pow(x\_entrada,2) -4;

return fx;

}

float ErrorRelativo(float ValorAnalitico, float ValorNumerico){

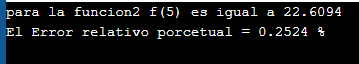
//epsilon= abs (ValorAnalitico-Valornumerico/ValorNumerico)\*100

float epsilon;

epsilon= fabs((ValorAnalitico - ValorNumerico)/ValorNumerico)\*100;

return epsilon;

}



**Método de bisección.**

include <stdio.h>

#include <math.h>

#define RAIZ(x) (log(x)) + pow(x,2) -4 // Definimos la funcion de la que se busca la raiz

int main() {

float a, b, xi, xa, Ea;

int iteraciones;

printf("Dame el valor de b: ");

scanf("%f", &b);

printf("Dame el valor de a: ");

scanf("%f", &a);

iteraciones = 0;

xa = 0.0;

Ea = 0;

printf("%12s %10s %10s %10s %10s\n","Iteraciones", "b", "a", "Xi","Error Aproximado");

do{

iteraciones++;

xi = (b + a) / 2;

if(iteraciones >= 0) {

Ea = ((xi - xa) / xi) \* 100; // Error aproximado

}

Ea = fabs(Ea); // valor absoluto del número

printf("%12d %10f %10f %10f %10f\n",iteraciones, b, a, xi, fabs(Ea));

xa = xi;

if(RAIZ(xi) > 0) {

b = xi;

} else{

a = xi;

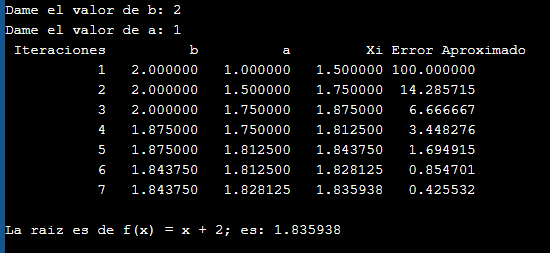
}

} while(Ea > 0.5 || (Ea == 0.0 && iteraciones == 1));

printf("\nLa raiz es de f(x) = x + 2; es: %f\n", xi);

return 0;

}



**Método de falsa posición.**

#include <stdio.h>

#include <math.h>

float

funcion2 (float x)

{

float y;

y = (log (x)) + pow (x, 2) - 4;

return y;

}

float

falsaposicion (float a, float b, int iter)

{

float xi, xi\_anterior = 0.0, errorRP;

int i = 1;

printf ("i \t a \t\t b \t\t xi \t\t Erp\n");

do

{

xi = b - ((funcion2 (b) \* (a - b)) / (funcion2 (a) - funcion2 (b)));

errorRP = ((xi - xi\_anterior) / xi) \* 100;

xi\_anterior = xi;

printf ("%d \t %.6f \t %.6f \t %.6f \t%.6f \n", i, a, b, xi,

fabs (errorRP));

if (funcion2 (a) \* funcion2 (xi) < 0)

{

b = xi;

}

else

{

a = xi;

}

i++;

}

while (funcion2 (a) \* funcion2 (xi) != 0 && i <= iter);

return xi;

}

int

main ()

{

float a = 1, b = 2, iteraciones = 5;

float raiz;

raiz = falsaposicion (a, b, iteraciones);

printf ("\n Raiz aproximada es %.6f", raiz);

printf ("\n\n");

return 0;

}

