Universidad Autónoma del Estado de Hidalgo
Instituto de Ciencias Básicas e Ingeniería
Licenciatura en Física y Tecnología Avanzada
Academia de Matemáticas y Física

Análisis Numérico

Presenta:
Palomares Maldonado Héctor Miguel

Enero - junio 2017

Universidad Autónoma del Estado de Hidalgo
Instituto de Ciencias Básicas e Ingeniería
Academia de Matemáticas y Física
optativa 1 (Análisis Numérico)

Palomares Maldonado Héctor Miguel

Índice

1. Aproximación numérica y errores

Dígitos significativos: Son dígitos que empiezan con el dígito distinto de cero del extremo izquierdo y terminan con el dígito correcto del extremo derecho incluyendo los ceros finales que son correctos.

Errores: Suponga que α y β son 2 números, en los cuales β es una aproximación de α.

El error de β como aproximación de α es: $\alpha - \beta$

El error absoluto de β como aproximación de α es : $|\alpha - \beta|$

El error relativo de β como aproximación de α es $\dfrac{|\alpha - \beta|}{|\alpha|}$

Ejemplo:

Fig. 1

Ejercicio:
x redondeada tiene 1 números significativos mientras que y redondeada tiene 3 el error absoluto de x es: 3×10^{-5} y el error relativo de x es: $8{,}6 \times 10^{-3}$ el error absoluto de u es: 2×10^{-3} y el error relativo de y es: $6{,}6x10^{-5}$

Exactitud: Exactitud a n cifras decimales significa que se puede confiar en n dígitos a la derecha del lugar decimal. Exactitud a n cifras significativas significa que se puede confiar en un total de n dígitos que sean importantes empezado con el dígito distinto de cero del extremo izquierdo.

Fig. 2

Redondeo: Decimos que x está redondeado a n dígitos cuando x se reemplaza por un n-dígito número que se aproxima a x con un error mínimo

Truncamiento: Un número x está truncado a n dígitos cuando todos los dígitos que siguen al n-ésimo dígito son descartados y ninguno de los n dígitos cambia.

Redondeo: Si tenemos un número con $n + 1$ dígitos y termina en 5, y lo queremos redondear a n dígitos, redondeamos hacia arriba? o hacia abajo?
Ejemplo

$$x = 7{,}35$$

Para un gran conjunto de datos, el redondeo parejo tiene a reducir el error total con una parte igual de números redondeados hacia arriba como redondeados hacia abajo
Una manera de hacer esto es que el n-ésimo dígito siempre sea un dígito par.
Ejemplos: Al redondear algunos números de 3 decimales a 2

$$0{,}217 \approx 0{,}22$$

$$0{,}365 \approx 0{,}36$$

$$0{,}475 \approx 0{,}48$$

$$0{,}592 \approx 0{,}59$$

MULTIPLICACIÓN ANIDADA

Fig. 3
Fig. 4
Fig. 5

Ejemplo:

$$10,587,431,831{,}23$$

$$105874 \times 10^5$$

Programa de límites de números de punto flotante

Fig. 6
Fig. 7

ejercicios en clase

programa 1

```c
#include <stdio.h>
int main()
{
    int i,n;
    float p,x,a[10];
    printf("calculo del polinomio \n\n");
    printf("grado del polinomio: ");
    scanf("%d",&n);
    printf("\n");
    for(i=0; i<=n; i++)
    {
        printf("escribe el elemento%d : ",i);
        scanf("%f",&a[i]);
    }
    p=a[n];
    printf("\nEscriba x :");
    scanf("%f",&x);
    printf("\n\na[%d]=%.2f\n\n",i,p);
    for(i=n-1; i>=0; i--)
    {
        p=a[i]+x*p;
        printf("%.2f\n",p);
    }
    printf("La aproximacion del polinomio es : %d \n", p);
    return 0;
}
```

```
ubuntu140@ubuntu:~/Documents/Programacion/AN$ gcc p1.c
ubuntu140@ubuntu:~/Documents/Programacion/AN$ ./a.out
calculo del polinomio

grado del polinomio: 4

escribe el elemento 0 : -2
escribe el elemento 1 : -5
escribe el elemento 2 : 7
escribe el elemento 3 : -4
escribe el elemento 4 : 1

Escriba x :3


a[5]=1.00

-1.00
4.00
7.00
19.00
La aproximacion del polinomio es : 19.000000
```

Programa 2.

```c
#include<stdio.h>
int main()
{
    int i,n;
    float p,x,a[10],c[10];
    printf("calculo del polinomio \n\n");
    printf("grado del polinomio: ");
    scanf("%d",&n);
    printf("\n");
    for(i=0; i<=n; i++)
    {
        printf("escribe el elemento %d : ",i);
        scanf("%f",&a[i]);
    }
    printf("\nEscriba x :");
    scanf("%f",&x);
    c[0]=0.0;

        for(i=0; i<=n; i++)
    {
        c[i+1]=a[i]+x*c[i];
        printf("%.2f\n",c[i+1]);
    }

    printf("La aproximacion del polinomio es : %f\n", c[i]);
    return 0;
}
```

```
ubuntu140@ubuntu:~/Documents/Programacion/AN$ gcc p2.c
ubuntu140@ubuntu:~/Documents/Programacion/AN$ ./a.out
calculo del polinomio

grado del polinomio: 4

escribe el elemento 0 : 1
escribe el elemento 1 : -4
escribe el elemento 2 : 7
escribe el elemento 3 : -5
escribe el elemento 4 : -2

Escriba x :3
1.00
-1.00
4.00
7.00
19.00
La aproximacion del polinomio es : 19.000000
```

2. Formula de Taylor

Problema 3 calcula :

1. La aproximación de e^x con polinomio de taylor
2. Mediante la función exponencial $exp()$ calcula e^x
3. error del polinomio de taylor como referencia la función $exp()$

```c
#include <stdio.h>
#include <math.h>
#define MAX 10
long factorial(int n)
{
    int i,j;
    j=1;
    for(i=1;i<=n;++i)
    {
        j=j*i;
    }
    return j;
}
int main()
{
        int i,n;
        double x,e=0,r;
        printf("SERIE DE TAYLOR DE LA FUNCION EXPONENCIAL: \n");
        printf("Introduzca el numero de terminos:");
        scanf(" %d",&n);

        printf("Introducir el valor de x a evaluar:");
        scanf(" %lf",&x);

        for (i=0;i<n;i++) {
            r=(pow(x, i))/(factorial(i));
            e=e+r;
        }
        printf("El resultado de la sere es: \n");
        printf("%lf",e);
        printf("El resultado aceptado es: \n");
        printf("%lf",exp(x));
        printf("El residuo de la operacion es: \n");
        printf("%lf",(exp(x)-e));
        return 0;
    }
```

programa 4
calcular la exponencial, e^x

```c
#include <stdio.h>
#include <math.h>
#define MAX 100
long factorial(int n)
{
    int i,j=1;
    for(i=1;i<=n;++i)
    {
        j=j*i;
    }
    return j;
}

int main()
{
    int i,n;
    double x,r2,r1,r3;
    r2=0;
    printf("SERIE DE TAYLOR DE LA FUNCIÒN EXPONENCIAL:\n");
    printf("Introducir el valor de x a evaluar:");
    scanf(" %lf",&x);
    printf("\nIntroduzca el residuo que desea:");
    scanf(" %lf",&r3);
    for (i=0;(exp(x)-r2)>=r3;i++)
    {
        r1=(pow(x, i))/(factorial(i));
        r2=r2+r1;
        printf("%d-\t r1 = %.5lf \t r2=%.5lf\n",i,r1,r2);
    }
    printf("Para el residuo %lf se utilizaron %d términos. \n", r3,i);
    printf("El resultado de la aproximación de la serie es: \n");
    printf("%lf",r2);
    printf("\nEl resultado de la serie \n");
    printf("%lf",exp(x));
    printf("\nEl residuo de la operación es %lf  \n\n\n",(exp(x)-r2));
   return 0;
}
```

```
ubuntu140@ubuntu:~/Documents/Programacion/AN$ gcc p4-2.c -lm
ubuntu140@ubuntu:~/Documents/Programacion/AN$ ./a.out
SERIE DE TAYLOR DE LA FUNCIÓN EXPONENCIAL:
Introducir el valor de x a evaluar:2

Introduzca el residuo que desea:0.9
0-       r1 = 1.00000    r2=1.00000
1-       r1 = 2.00000    r2=3.00000
2-       r1 = 2.00000    r2=5.00000
3-       r1 = 1.33333    r2=6.33333
4-       r1 = 0.66667    r2=7.00000
Para el residuo 0.900000 se utlizaron 5 términos.
El resultado de la aproximacion de la serie es:
7.000000
El resultado de la serie
7.389056
El residuo de la operación es 0.389056
```

Problema 5
programa que calcula la serie de Taylor de $\ln(x+1)$ sabemos que

$$\ln(x+1) = x - \frac{x^2}{2} + \frac{x^3}{3} + \frac{x^4}{4} + \ldots$$

```c
#include <stdio.h>
#include <math.h>
#define MAX 100
int main() {
    int i,n,signo;
    double x,ln,r1;
     printf("SERIE DE TAYLOR DE LA FUNCION ln(x+1)\n");
    printf("Introducir el valor de x a evaluar:");
    scanf(" %lf",&x);
    printf("¿cuantos Terminos? : ");
    scanf("%d",&n);

    printf("\n\ntermino \t r1 \t ln\n");
    for (i=1; i<=n ;i++)
    {
        signo=pow((-1),i+1);
        r1=  (signo)*(pow(x,i)/i);
        ln=ln + r1;
        printf(" %d \t %lf \t %lf \n",i,r1,ln);
    }
    printf("\nLa aproximacion es %lf",ln );

    return 0;
}
```

```
ubuntu@ubuntu:~/Documents/Programacion/AN$ gcc ln.c -lm
ubuntu@ubuntu:~/Documents/Programacion/AN$ ./a.out
SERIE DE TAYLOR DE LA FUNCION ln(x+1)
Introducir el valor de x a evaluar:0.2
¿cuantos Terminos? : 10


termino          r1      ln
 1       0.200000        0.200000
 2       -0.020000       0.180000
 3       0.002667        0.182667
 4       -0.000400       0.182267
 5       0.000064        0.182331
 6       -0.000011       0.182320
 7       0.000002        0.182322
 8       -0.000000       0.182322
 9       0.000000        0.182322
 10      -0.000000       0.182322

La aproximacion es 0.182322
```

Problema 6
programa que calcula la serie de Taylor de $\ln(x+1)$

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

int main() {
    int i,signo=1;
    long j,n;
    double r,ln=0,pot=1,x;
    printf("SERIE DE TAYLOR DE LA FUNCION ln(x+1)\n");
    printf("Introducir el valor de x a evaluar :  ");
    scanf(" %lf",&x);
    printf("¿cuantos Terminos? : ");
    scanf("%d",&n);
    printf("\n\ntermino \t r1 \t ln\n");
    for (i=1; i<=n ;i++)
    {
        pot=1;
        for(j=1; j<=i; j++)
        {
            pot*=x;
        }
        signo*=(-1);
        r= -(signo)*((1.0*pot)/i);
        ln=ln + r;
        printf(" %d \t %lf \t %lf \n",i,r1,ln);
    }
    printf("\nLa aproximacion es %lf\n",ln );

    return 0;
```

```
ubuntu@ubuntu:~/Documents/Programacion/AN$ ./a.out
SERIE DE TAYLOR DE LA FUNCION ln(x+1)
Introducir el valor de x a evaluar :  0.2
¿cuantos Terminos? : 10


termino          r       ln
1        0.200000        0.200000
2        -0.020000       0.180000
3        0.002667        0.182667
4        -0.000400       0.182267
5        0.000064        0.182331
6        -0.000011       0.182320
7        0.000002        0.182322
8        -0.000000       0.182322
9        0.000000        0.182322
10       -0.000000       0.182322

La aproximacion es 0.182322
```

Universidad Autónoma del Estado de Hidalgo
Instituto de Ciencias Básicas e Ingeniería
Academia de Matemáticas y Física
optativa 1 (Análisis Numérico)

Examen primer parcial

Palomares Maldonado Héctor Miguel

```
1
2  #include<stdio.h>
3  #include<math.h>
4
5  void expon();
6  void LN();
7  void seno();
8  void cosenov();
9  long factorial (int n);
10
11
12 int main()
13 {
14     int opc;
15     menu:
16     printf("Programa que que calcula las series de Taylor de:\n 1) e^x\n
           2) ln(x)\n 3) sen(x)\n 4) cos(x)\n\nEliga una opcion:   ");
17     scanf("%d",&opc);
18     switch(opc)
19     {
20         case 1:
21                 expon();
22                 goto menu;
23                 break;
24         case 2:
25                 LN();
26                 goto menu;
27
28         case 3:
29                 seno();
30                 goto menu;
31
32         case 4:
33                 coseno();
34                 goto menu;
35
36         default:
37             printf("opcion NO valida....\n");
38             break;
39     }
40     return 0;
41 }
42
43 long factorial (int n)
44 {
45 int i,factorial=1;
46     for(i=1;i <=n;++i)
47     {
48         factorial=factorial*i;
```

```
49      }
50      return factorial;
51  }
52
53
54  void expon()
55  {
56      int i,n;
57      double x,e=0 ,termino ,error;
58      printf (" SERIE DE TAYLOR DE LA FUNCION EXPONENCIAL :\n");
59      printf (" Introducir el valor de x :");
60      scanf (" %lf" ,&x);
61      printf ("\ nIntroduzca el residuo que desea :");
62      scanf (" %lf" ,&error);
63      for (i=0;( exp(x)-e) >=error;i++)
64      {
65          termino=( pow (x, i))/( factorial (i));
66          e=e+termino;
67          printf (" %d -\t r1 = %.5lf \t r2 = %.5lf\n",i,termino ,e);
68      }
69      printf (" Para el error %lf se utilizaron %d terminos . \n", error ,i)
            ;
70      printf ("El resultado de la aproximacion de la serie es: %lf \n",e);
71      printf ("\nEl resultado de la serie %lf\n", exp(x));
72  }
73
74
75  void LN()
76  {
77      int i,n,signo;
78      double x,ln=0,termino;
79       printf("SERIE DE TAYLOR DE LA FUNCION ln(x+1)\n");
80      printf("Introducir el valor de x a evaluar:");
81      scanf(" %lf", &x);
82      printf("¿cuantos Terminos? : ");
83      scanf("%d", &n);
84
85      printf("\n\nnum term. \t Termino \t ln \n");
86      for (i=1; i<=n ;i++)
87      {
88          signo=pow((-1),i+1);
89          termino = (signo)*(pow(x,i)/i);
90          ln=ln+termino;
91          printf("    %d \t\t %.6lf \t %.5lf \n", i, termino, ln);
92      }
93      printf("\nLa aproximacion es %lf\n\n\n\n",ln );
94
95  }
96
97
98  void seno()
99  {
100     int i,j=3,n,signo;
101     double x,aprox,termino;
102     printf (" SERIE DE TAYLOR DE LA FUNCION sen(x) \n");
103     printf (" teclear el valor de x :");
104     scanf (" %lf" ,&x);
105     printf("introduzca el numero de terminos : ");
```

```
106     scanf("%d",&n);
107     printf("\n interacion \t c/u-term. \t aprox.\n ");
108 
109             aprox=x;
110             for (i=1; i<=n; i++)
111             {
112                 signo=pow((-1),i);
113                 termino=(signo)*(pow(x,j)/factorial(j));
114                 aprox=aprox+termino;
115                 j=j+2;
116                 printf ("   %d- \t\t %.7lf \t %.7lf\n",i,termino ,aprox);
117             }
118         printf("El resultado de la aproximacion de la serie es : %lf \n\n"
                ,aprox);
119 }
120 
121 void coseno()
122 {
123     int i,j=2,n,signo;
124     double x, aprox=1,termino;
125     printf (" SERIE DE TAYLOR DE LA FUNCION cos(x) \n");
126     printf (" teclear el valor de x :");
127     scanf (" %lf" ,&x);
128     printf("introduzca el numero de terminos : ");
129     scanf("%d",&n);
130     printf("\n interacion \t c/u-term. \t aprox.\n ");
131 
132             for (i=1; i<=n; i++)
133             {
134                 signo=pow((-1),i);
135                 termino=(signo)*(pow(x,j)/factorial(j));
136                 j=j+2;
137                 aprox=aprox+termino;
138 
139                 printf ("   %d- \t\t %.7lf \t %.7lf\n",i+1,termino ,aprox)
                        ;
140             }
141         printf("El resultado de la aproximacion de la serie es : %lf\n\n\n
                ",aprox);
142 }
```

3. Localización de raices de ecuaciones

Un numero real o complejo r para el que $f(r) = 0$ se llama raíz o cero de f, por ejemplo

$$f(x) = 6x^2 - 7x + 2$$

tienen $1/2$ y $3/2$ como ceros

$$6\left(\frac{1}{2}\right)^2 - 7\left(\frac{1}{2}\right) + 2 \qquad 6\left(\frac{2}{3}\right)^2 - 7\left(\frac{2}{3}\right) + 2$$

Algunas funciones son demasiado complicadas para calcular los ceros de manera manual, por ejemplo:
$f(x) = 3{,}24x^8 - 2{,}42x^7 + 10{,}34x^6 + 11{,}01$
$g(x) = 2^{x2} - 10x * 1$
$h(x) = \cosh\left(\sqrt{x^2+1} - e^x\right) + \ln|\operatorname{sen} x|$

3.1. Método de Bisección

Empezamos con un intervalo $[a, b]$ donde $f(a)$ y $f(b)$ tienen signo contrario
Calculemos la mitad del intervalo

$$c = \frac{a+b}{2}$$

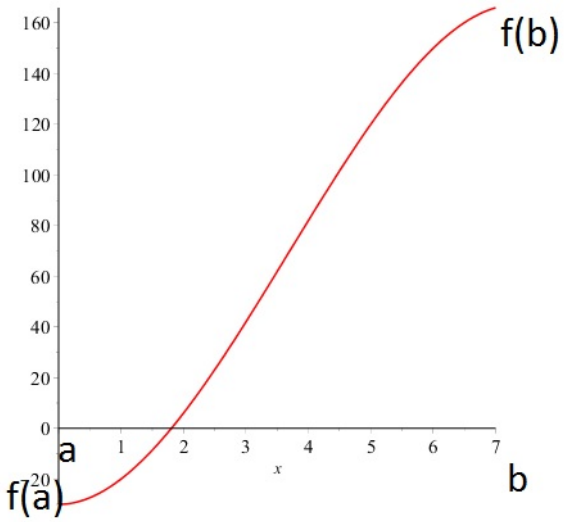
si $f(c)$ tiene el mismo signo que $f(a)$

$$a = c$$
$$f(a) = f(c)$$

si $f(c)$ tiene el mismo signo que $f(b)$

$$b = c$$
$$f(b) = f(c)$$

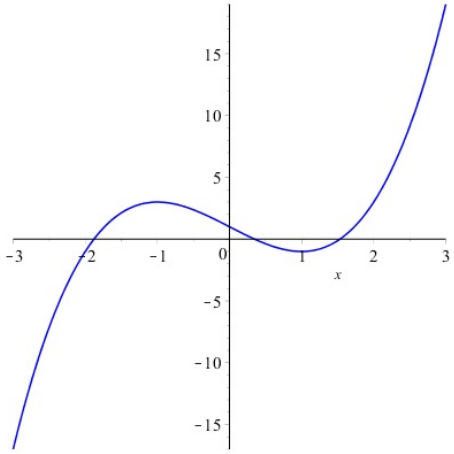
si $f(c) = 0$ ya terminaron



ejercicio

probar que $f(x) = x^3 - 3x + 1$ en el intervalo $[0, 1]$

n	a	b	c	f(c)	error	f(a)	f(b)
0	0	1	0.5	-0.375	05	1	-1
1	0	0.5	0.25	0..2656	0.25	1	-0.375
2	0.25	0.5	0.375	-0.0722	0.125	0.2656	-0.375
3	0.25	0.375	0.3125	0.0930	0.0625	0.2656	-0.0722
4	0.3125	0.375	0.34375	0.00956	0.03125	0.0930	-0.0722

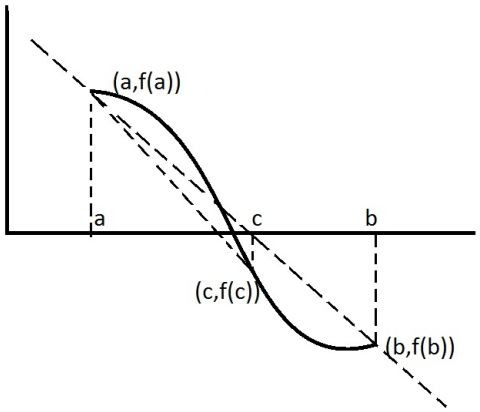



```c
#include<stdio.h>
#include<math.h>
#define INTERVALOS 10
void tabla(double a, double b);
double f(double x);
int main()
{
    double a,b,c,pmed,x,l,n_t,error;
    int i,n;
    printf("Calculo de las raices de una funcion aplicando el metodo
     de biseccion\n");
    printf("Ingrese el intervalo [a,b], Para encoontrar el cambio
     de signo...\n");
    printf("a = ");
    scanf("%lf",&a);
    printf("b = ");
    scanf("%lf",&b);
    tabla(a,b);  //muestra la tabla de x y f(x)
    printf("Introducir numero de terminos : ");
    scanf("%d", &n);
    printf("Introducir el error deseado : ");
    scanf("%lf", &error);
    printf("\n\t INTRODUCIR EL INTERVALO \n");
    printf("a = ");
    scanf("%lf",&a);
    printf("b = ");
    scanf("%lf", &b);

    if(f(a) * f(b) > 0)
     printf("No se puede aplicar el metodo de la biseccion\n
     porque f(%lf) y f(%lf) tienen el mismo signo \n",a,b);
```

```
32       else
33       {
34          if(f(a)*f(b)<0)
35          {
36               n_t=(log(b-a)-log(2*error))/(log(2));
37               printf("\n  n \t\t a \t\t b \t\t pmed \t\t f(c) \t\t
38               f(a) \t\t f(b) \t error\n\n");
39               for(i=0;i<n;i++)
40               {
41                    pmed=(a+b)/2;
42                    printf(" %d \t %lf \t %lf \t %lf \t %lf \t %lf \t
43                     %lf \t %lf  \n", i, a, b, pmed, f(pmed),
44                      f(a), f(b), fabs(pmed-a));
45                    if(fabs(b-a)<=error)
46                       {
47                              break;
48                       }
49                    if(f(pmed)==0)
50                    {
51                       break;
52                    }
53                     if(f(pmed)*f(a)>0)
54                    {
55                       a=pmed;
56                   }
57                   if(f(pmed)*f(b)>0){
58                       b=pmed;
59                   }
60               }//fin for
61
62           }
63      }//else
64      printf("\n\nPara el error %lf  o %d terminos\n\nLa raiz es  : %lf\n\n\
           n\n ",error,n,pmed);
65  return 0;
66  }///Fin de Main()
67  void tabla(double a, double b)
68  {
69      int i,puntos;
70      double ancho;
71      puntos=INTERVALOS+1;
72      ancho=(b-a)/(INTERVALOS);
73      printf("\t x \t f(x)\n");
74      for(i=0; i<puntos; i++ )
75      {
76          printf("\t %.2lf \t %.2lf\n",a,f(a));
77          a=a+ancho;
78      }
79  }
80  double f(double x)
81  {
82      double f;
83      f=pow(x,3)-3*x+1;
84      return f;
85  }
```

```
ubuntu@ubuntu:~/Documents/Programacion/AN$ gcc p6.c -lm
ubuntu@ubuntu:~/Documents/Programacion/AN$ ./a.out
Calculo de las raices de una funcion aplicando el metodo de biseccion
Ingrese el intervalo [a,b], Para encoontrar el cambio de signo...
a = 0
b = 3
        x       f(x)
        0.00    1.00
        0.30    0.13
        0.60    -0.58
        0.90    -0.97
        1.20    -0.87
        1.50    -0.12
        1.80    1.43
        2.10    3.96
        2.40    7.62
        2.70    12.58
        3.00    19.00
Introducir numero de terminos : 25
Introducir el error deseado : 0.001

        INTRODUCIR EL INTERVALO
a = 0.3
b = 0.6

  n             a               b              pmed           f(c)           f(a)           f(b)    error

 0       0.300000        0.600000        0.450000       -0.258875       0.127000       -0.584000       0.150000
 1       0.300000        0.450000        0.375000       -0.072266       0.127000       -0.258875       0.075000
 2       0.300000        0.375000        0.337500        0.025943       0.127000       -0.072266       0.037500
 3       0.337500        0.375000        0.356250       -0.023537       0.025943       -0.072266       0.018750
 4       0.337500        0.356250        0.346875        0.001112       0.025943       -0.023537       0.009375
 5       0.346875        0.356250        0.351562       -0.011236       0.001112       -0.023537       0.004687
 6       0.346875        0.351562        0.349219       -0.005068       0.001112       -0.011236       0.002344
 7       0.346875        0.349219        0.348047       -0.001979       0.001112       -0.005068       0.001172
 8       0.346875        0.348047        0.347461       -0.000434       0.001112       -0.001979       0.000586
 9       0.346875        0.347461        0.347168        0.000339       0.001112       -0.000434       0.000293


Para el error 0.001000  o 25 terminos

La raiz es : 0.347168
```

3.2. Método de la falsa posición



si $f(c)$ tiene el mismo signo que $f(b)$, entonces $b = c$
si $f(c)$ tiene el mismo signo que $f(a)$, entonces $a = c$
Trazamos una recta entre los puntos

$$(a, f(a)) \qquad (b, f(b))$$

El punto c se encuentra en la intersección de la recta con el eje x

Para obtener c, usando triángulos semejantes :

$$\frac{b-c}{-f(b)} = \frac{c-a}{f(a)}$$

donde

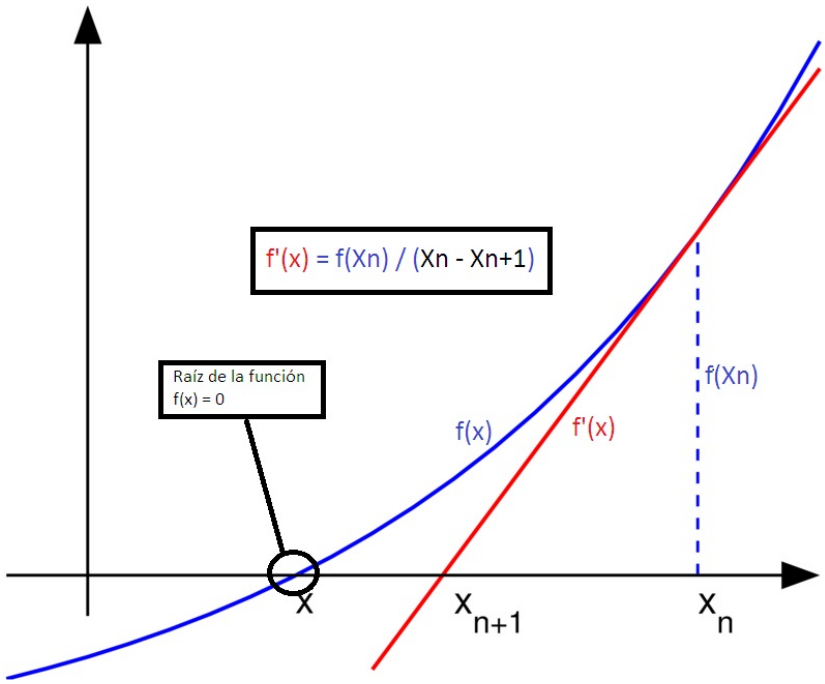
$$c = \frac{af(a) - bf(b)}{f(b) - f(a)}$$

```c
#include<stdio.h>
#include<math.h>
#define INTERVALOS 10
void tabla(double a, double b);
double f(double x);
int main()
{
    double a,b,c,pmed,x,l,n_t,error,d=0,f_a,f_b,f_c;
    int i,n,g,h=2;
    printf("Calculo de las raices de una funcion aplicando el metodo de biseccion\n");
    printf("Ingrese el intervalo [a,b], Para encoontrar el cambio de signo ...\n");
    printf("a = ");
    scanf("%lf",&a);
    printf("b = ");
    scanf("%lf",&b);
    tabla(a,b);  //muestra la tabla de x y f(x)
    printf("Introducir numero de terminos : ");
    scanf("%d", &n);
    printf("Introducir el error deseado : ");
    scanf("%lf", &error);
    printf("\n\t INTRODUCIR EL INTERVALO \n");
    printf("a = ");
    scanf("%lf",&a);
    printf("b = ");
    scanf("%lf", &b);
                f_a=f(a);   f_b=f(b);
    if(f(a) * f(b) > 0)
     printf("No se puede aplicar el metodo de la biseccion\nporque f(%lf) y f(%lf) tienen el mismo signo \n",a,b);
    else
    {
    if(f(a)*f(b)<0)
    {
    printf("\n  n \t\t a \t\t b \t\t pmed \t\t f(c) \t\t f(a) \t\t f(b) \t\t error \n\n");
        for(i=0;i<n;i++)
        {
         pmed=((a*f_b)-( b*f_a))/(f_b- f_a);
         f_c=f(pmed);
         if(fabs(b-a) < error)
        {
            break;
        }//fin del  if
        else
        {
          printf(" %d \t %lf \t %lf \t %lf \t %lf \t %lf \t %lf \t ", i, a, b, pmed, f_c, f_a, f_b);
        if(f_c*f_a>0)
        {
         a=pmed;
         g=1;
        }
        if(f_c*f_b>0)
        {
          b=pmed;
          g=0;
```

```
54           }
55          if(g==0 && h==0)
56          {
57             f_a=f_a/2;
58          }
59          if(g==1 && h==1)
60          {
61             f_b=f_b/2;
62          }
63           printf(" \t%lf \n",fabs(pmed) - d);
64           d=pmed;
65           h=g;
66          }//fin for
67          }//fin else
68           }fin if
69      }//else
70      printf("\n\nPara el error %lf  o %d terminos\n\nLa raiz es : %lf\n\n\n
            \n ",error,n,pmed);
71  return 0;
72  }///Fin de Main()
73  void tabla(double a, double b)
74  {
75      int i,puntos;
76      double ancho;
77      puntos=INTERVALOS+1;
78      ancho=(b-a)/(INTERVALOS);
79      printf("\t x \t f(x)\n");
80      for(i=0; i<puntos; i++ )
81      {
82          printf("\t %.2lf \t %.2lf\n",a,f(a));
83          a=a+ancho;
84      }
85  }
86  double f(double x)
87  {
88      double f;
89      f=pow(x,3)-3*x+1;
90      return f;
91  }
```

```
ubuntu@ubuntu:~/Documents/Programacion/AN$ gcc p7.c -lm
ubuntu@ubuntu:~/Documents/Programacion/AN$ ./a.out
Calculo de las raices de una funcion aplicando el metodo de biseccion
Ingrese el intervalo [a,b], Para encoontrar el cambio de signo...
a = 0
b = 3
        x       f(x)
        0.00    1.00
        0.30    0.13
        0.60    -0.58
        0.90    -0.97
        1.20    -0.87
        1.50    -0.12
        1.80    1.43
        2.10    3.96
        2.40    7.62
        2.70    12.58
        3.00    19.00
Introducir numero de terminos : 20
Introducir el error deseado : 0.001

        INTRODUCIR EL INTERVALO
a = 0.3
b = 0.6

  n             a               b              pmed           f(c)           f(a)           f(b)           error

 0       0.300000        0.600000        0.353586        -0.016553       0.127000        -0.584000       0.353586
 1       0.300000        0.353586        0.309572        0.100953        0.127000        -0.584000       -0.044015
 2       0.309572        0.353586        0.317434        0.079685        0.127000        -0.584000       0.007862
 3       0.317434        0.353586        0.328392        0.050239        0.127000        -0.292000       0.010958
 4       0.328392        0.353586        0.340112        0.019006        0.127000        -0.146000       0.011721
 5       0.340112        0.353586        0.348668        -0.003618       0.127000        -0.073000       0.008556
 6       0.340112        0.348668        0.345545        0.004622        0.127000        -0.073000       -0.003123
 7       0.345545        0.348668        0.347529        -0.000613       0.127000        -0.073000       0.001983
 8       0.345545        0.347529        0.346805        0.001297        0.127000        -0.073000       -0.000724


Para el error 0.001000  o 20 terminos

La raiz es : 0.347264
```

3.3. Método de Newton-Raphson



$$x_i = x_{i-1} - \frac{f(x_{i-1})}{f('(x_{i-1})}$$

pendiente

$$f'(x_0) = \frac{f(x)}{x_0 - x_1}$$

error

$$\frac{f(x_{i-1})}{f('(x_{i-1})}$$

```c
#include<stdio.h>
#include<math.h>
#include<stdbool.h>
#define INTERVALOS 10
void tabla(double a, double b, int intervalos);
double f(double x);
double der_f(double x);

int main()
{   int intervalos=1,maxInt;
    double a,b,error,error1=0,x0,x1;
    bool converge = true;
    printf("\tCalculo de raices - Metodo de Newton-Raphson\n");
    printf("Ingrese el intervalo [a,b], Para ver si existe un cambio de
       signo\n");
    printf("a : "); scanf("%lf", &a);
    printf("b : "); scanf("%lf", &b);
    tabla(a,b,INTERVALOS);
    printf("Introducir el error deseado : ");
    scanf("%lf", &error);
    printf("\n\t INTRODUCIR EL PUNTO INICIAL ADECUADO \n");
    printf("x = ");
    scanf("%lf", &x0);
    printf("Numero maximo de interacciones : ");
    scanf("%d",maxInt);
    printf("interaccion \t punto \t\tf(x) \t\t  df(x)/dx \t\t error \n");
    do
    {
        if(intervalos>maxInt)
        {
            converge = false;
            break;
         }
         else
         {
             x1=x0-f(x0)/der_f(x0);
             error1=fabs(x1-x0);
             printf("    %d \t\t %.8lf \t %.8lf \t %.8lf \t    %.8lf\n",
                intervalos,x0,f(x0),der_f(x0), error1);
             if(error1<=error)// termina
             {
                converge=true;
                 break;
             }
             else
             {
                x0=x1;
                intervalos++;
             }
         }
    }while(1);
        printf("Para el  error de %lf la raiz es  %lf: ", error, x1 );
   return 0;
}
void tabla(double a, double b, int intervalos)
{
    int i,puntos;
    double ancho;
```

```
57      puntos=intervalos+1;
58      ancho=(b-a)/(intervalos);
59      printf("\t x \t f(x)\n");
60      for(i=0; i<puntos; i++ )
61      {
62          printf("\t %.2lf \t %.2lf\n",a,f(a));
63          a=a+ancho;
64      }
65  }
66  double f(double x)
67  {
68      double f;
69      f=pow(x,3)-3*x+1;
70      return f;
71  }
72
73  double der_f(double x)
74  {
75      double df;
76      df=(3*pow(x,2))-3;
77      return df;
78  }
```

```
ubuntu@ubuntu:~/Documents/Programacion/AN$ gcc p8.c -lm
ubuntu@ubuntu:~/Documents/Programacion/AN$ ./a.out
       Calculo de raices - Metodo de Newton-Raphson
Ingrese el intervalo [a,b], Para ver si existe un cambio de  signo
a : 0
b : 3
         x       f(x)
         0.00    1.00
         0.30    0.13
         0.60    -0.58
         0.90    -0.97
         1.20    -0.87
         1.50    -0.12
         1.80    1.43
         2.10    3.96
         2.40    7.62
         2.70    12.58
         3.00    19.00
Introducir el error deseado : 0.00001

         INTRODUCIR EL PUNTO INICIAL ADECUADO
x = 0.6
Numero maximo de interacciones : 12

interaccion      punto          f(x)             df(x)/dx              error
    1           0.60000000      -0.58400000     -1.92000000        0.30416667
    2           0.29583333      0.13839055      -2.73744792        0.05055459
    3           0.34638792      0.00239745      -2.64004622        0.00090811
    4           0.34729603      0.00000086      -2.63815640        0.00000033

Para el  error de 0.000010 la raiz es  0.347296
```

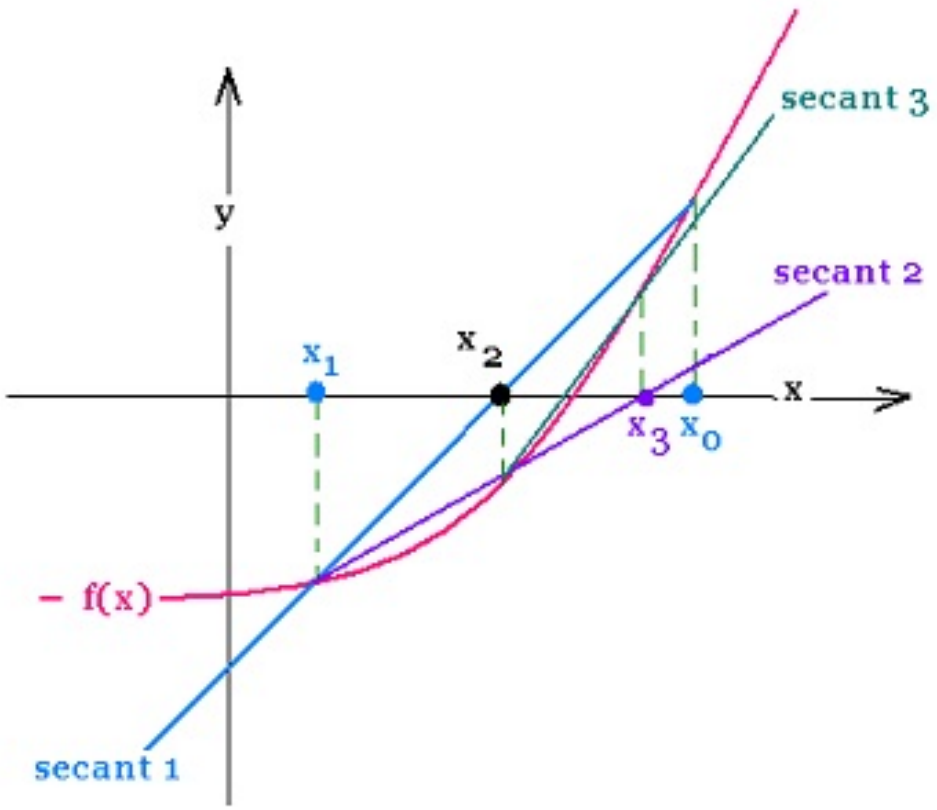
3.4. Metodo de la Secante

este metodo inmita al metodo de newton, pero evita el calculo de derivadas
La recta que pasa por x_0 y x_1 es una aproximacion a la tangente de $f(x)$ es x_1 (derivada)

$$f(x_1) \approx \frac{f(x_0) - f(x_1)}{x_0 - x_1}$$

$$f(x_1) \approx \frac{f(x_{n-1}) - f(x_1)}{x_{n-1} - x_1}$$

$$x_{n+1} = x_n - \left[\frac{x_n - x_{n-1}}{f(x_n) - f(x_{n-1})}\right] f(x_n)$$




```
1
2  #include<stdio.h>
3  #include<stdio.h>
4  #include<math.h>
5  #define INTERVALOS 10
6  void tabla(double a, double b);
7  double f(double x);
8  double sec(double a, double b);
9  int main()
10 {
11     double a,b,x,y,error,f_x;
12     int i,n;
13     printf("Calculo de las raices de una funcion aplicando el metodo de la
           secante\n");
14     printf("Ingrese el intervalo [a,b], Para encoontrar el cambio de signo
           ...\n");
15     printf("a = ");
16     scanf("%lf",&a);
17     printf("b = ");
```

```
18      scanf("%lf",&b);
19      tabla(a,b);  //muestra la tabla de x y f(x)
20      printf("Introducir numero de terminos : ");
21      scanf("%d", &n);
22      printf("Introducir el error deseado : ");
23      scanf("%lf", &error);
24      printf("\n\t INTRODUCIR EL INTERVALO \n");
25      printf("a = ");
26      scanf("%lf",&a);
27      printf("b = ");
28      scanf("%lf", &b);
29      x=a;   y=b; f_x=f(b);
30      printf("\n  Terminos \t x \t\t f(x) \t\t sec \t\t error\n");
31      for(i=0; i<n; i++)
32      {
33          printf("       %d  \t %.6lf  \t  %.6lf \t  %.6lf  ",i, b, f(b),
                sec(a,b));
34              if(i==0)printf(" \t 0.00000\n");
35              if(i != 0) printf("\t %lf \n",fabs(f(b)/sec(b,y)));
36              if(fabs(f(b)/sec(a,b))<=error)
37              {
38                  break;
39              }
40              y=b;   b=b-(f(b)/sec(a,b));
41              if(b>a)
42              {
43                  x=a, a=b; b=x;
44              }
45      }
46      printf("\n\nPara el error %lf  o %d terminos\n\nLa raiz es : %lf\n\n\n
            \n ",error,n,x);
47  return 0;
48  }///Fin de Main()
49  void tabla(double a, double b)
50  {
51      int i,puntos;
52      double ancho;
53      puntos=INTERVALOS+1;
54      ancho=(b-a)/(INTERVALOS);
55      printf("\t x \t f(x)\n");
56      for(i=0; i<puntos; i++ )
57      {
58          printf("\t %.2lf \t %.2lf\n",a,f(a));
59          a=a+ancho;
60      }
61  }
62  double f(double x)
63  {
64      double f;
65      f=pow(x,3)-3*x+1;
66      return f;
67  }
68  double sec(double x, double y)
69  {
70      double s;
71      s = (f(y)-f(x))/(x-y);
72      return s;
73  }
```

```
ubuntu@ubuntu:~/Documents/Programacion/AN$ gcc p9.c -lm
ubuntu@ubuntu:~/Documents/Programacion/AN$ ./a.out
Calculo de las raices de una funcion aplicando el metodo de la secante
Ingrese el intervalo [a,b], Para encoontrar el cambio de signo...
a = 0
b = 3
        x       f(x)
        0.00    1.00
        0.30    0.13
        0.60    -0.58
        0.90    -0.97
        1.20    -0.87
        1.50    -0.12
        1.80    1.43
        2.10    3.96
        2.40    7.62
        2.70    12.58
        3.00    19.00
Introducir numero de terminos : 20
Introducir el error deseado : 0.001

        INTRODUCIR EL INTERVALO
a = 1.5
b = 1.8

  Terminos       x               f(x)            sec             error
       0         1.800000         1.432000        -5.190000       0.00000
       1         1.500000         -0.125000       -6.673297       0.024085
       2         1.481269         -0.193670       -6.578569       0.052828
       3         1.451829         -0.295311       -6.431106       0.085535
       4         1.405910         -0.438833       -6.204556       0.140403
       5         1.335182         -0.625301       -5.863862       0.237176
       6         1.228546         -0.831363       -5.369107       0.430229
       7         1.073704         -0.983303       -4.691183       1.002081
       8         0.864098         -0.947102       -3.849882       5.483774
       9         0.618090         -0.618137       -2.974560       0.462258
       10        0.410282         -0.161782       -2.329465       0.073670
       11        0.340831         0.017099        -2.133127       0.006639
       12        0.348847         -0.004089       -2.155296       0.001547
       13        0.346950         0.000914        -2.150037       0.000346


Para el error 0.001000  o 20 terminos

La raiz es : 1.500000
```

4. **Solución numérica de ecuaciones algebraicas**
5. **Solución numérica de sistemas de ecuaciones lineales y no lineales.**
6. **Interpolación y aproximación polinomial.**
7. **Diferenciación e integración numérica.**
8. **Solución numérica de ecuaciones diferenciales ordinarias.**