UNIVERSIDAD CENTRO OCCIDENTAL LISANDRO ALVARADO
DECANATO DE CIENCIAS Y TECNOLOGIA
PROGRAMA DE LICENCIATURA EN CIENCIAS MATEMATICAS
DEPARTAMENTO DE MATEMÁTICAS
BARQUISIMETO

APUNTES DE CÁLCULO NUMÉRICO
(UNIDAD II)

CALCULO DE RAICES A ECUACIONES NO LINEALES

PEDRO ENRIQUE ADAMES ROJAS
ENERO, 2001

2.1. INTRODUCCIÓN

En esta Unidad se desarrollarán métodos numéricos para encontrar un “cero” ó “raíz” a una ecuación, lo cual es de suma importancia desde dos puntos de vista: Primero porque las ecuaciones modelan fenómenos que ocurren en la vida real y que a través de ellas damos solucion a las interrogantes que ellas plantean. Segundo, porque no siempre es posible encontrar tales raíces mediante procedimientos algebraicos y tampoco mediante procedimientos gráficos, ya que estos últimos por lo general no nos permiten establecer la aproximación a la raíz con un grado de satisfacción adecuado (una cantidad suficiente de dígitos). Por otra parte, los procedimientos gráficos suelen ser lentos si no se disponen de herramientas computacionales.

2.2. CALCULO DE RAICES

Como ya se indico, en ésta unidad se dará solución a ecuaciones de la forma $f(x)=0;$ es decir se encontrarán valores de x tales que $f(x)=0;$ por lo que a tales soluciones se les llama "ceros" o "raíces" de la ecuación. Cuando se trata el tema de cálculo de raíces o ceros de una ecuación, pueden enfocarse dos tipos de problemas :

- Calculo de raíces reales de ecuaciones algebraicas y transcendentes.
- Cálculo de todas las raíces reales y complejas de un polinomio.

En éste curso, el estudio será para los problemas indicados en primera instancia. Los métodos para encontrar raíces a éste tipo de problemas, se pueden clasificar según el hecho de apoyarse o no directamente en el intervalo en donde existe la raíz .entre los primeros están: Métodos Gráficos, Bisección y Regla Falsa; y entre los del segundo tipo (abiertos) se tienen entre otros, los Métodos de: Iteración al Punto Fijo, Newton-Raphson y Secante. Aunque en éste curso no se estudiarán todos estos métodos, los que se desarrollaran reciben el nombre de métodos iterativos, en donde interpretaremos que cada iteración es un paso dado hacia la búsqueda de la solución.

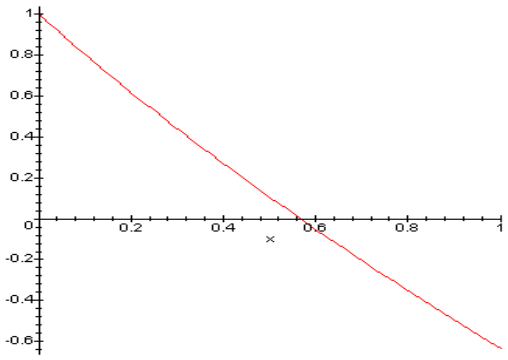
2.2.1. Métodos Gráficos

Para ilustar éste procedimiento se utilizará el paquete Matemático MAPLE V (® por Waterloo Maple Inc., Ontario – Canadá) para Windows. Supongamos que deseamos encontrar las raíces a la ecuación $f(x) = e^{-x} - x.$ Procedemos entonces de la manera siguiente, desde la forma de un nuevo archivo ingresamos los comandos indicados en

cada "promt", obteniendo como repuesta de cada uno de ellos la figura en la línea siguiente, así:

```
> p:= exp(-x) - x;
                    p := exp(-x) - x
> f:=unapply(p,x);
                 f := x -> exp(-x) - x
> f(0.0);
                         1
> f(0.2);
                      .6187307531
> f(0.4);
                      .2703200460
> f(0.6);
                      -.0511883639
> f(0.8);
                      -.3506710359
> f(1.0);
                      -.6321205588
```

- plot(f(x),x=0..1.0);



De éste gráfico se observa que aproximadamente la raíz es 0.57, veamos cual es su imagen:

```
> f(0.57);
                      -.0044745613
```

Se observa que los métodos gráficos por lo general no son precisos; como ya se había establecido, sin embargo son útiles para obtener aproximaciones de la raíz y al utilizar computadoras, estos métodos adquieren gran importancia.

Se pueden generalizar los métodos gráficos, en cuanto a la posesión de raíces de la manera siguiente :

Sean $x_l, x_u \in [a,b], f$ una función continua en $[a,b]$:

a) Si $f(x_l)$ y $f(x_u)$ tienen el mismo signo, entonces no hay raíces sobre él, o hay un número par de ellos.

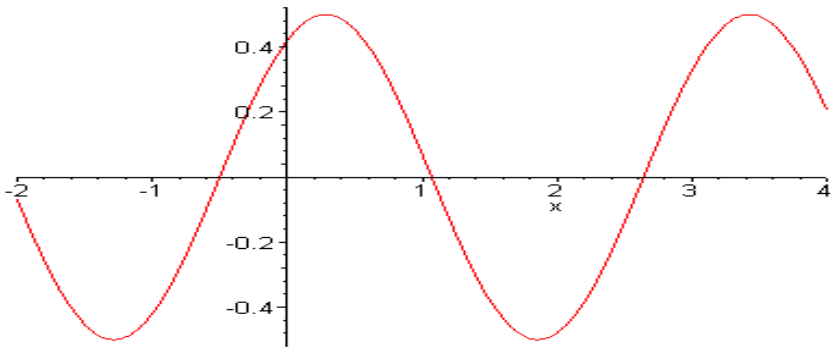
b) Si $f(x_l)$ y $f(x_u)$ tienen signos contrarios, entonces generalmente hay un número impar de raíces dentro del intervalo.

c) Pueden ocurrir raíces múltiples cuando la función es tangencial al eje X.

d) Las funciones discontinuas en donde los extremos del intervalo tienen imágenes de signos contrarios, tienen un número par de raíces.

2.2.2. Método y Algoritmo de Bisección

El método de Bisección ó búsqueda binaria o método de Bolzano, se apoya en el Teorema del Valor Intermedio, el cual establece lo siguiente: "Si $f \in C$ $[a,b]$ (continua en $[a,b]$) y k es cualquier número entre $f(a)$ y $f(b)$, entonces existe $c \in (a,b)$ tal que $f(c)=k$". Esto es: Si $f(a)$ y $f(b)$ son de signos contrarios para la función continua f en $[a,b]$, entonces por el teorema anterior existe $p \in (a,b)$ tal que $f(p)=0$ ya que al ser $f(a)$ y $f(b)$ de signos contrarios. $0 \in [f(a),f(b)]$ ó $0 \in [f(b),f(a)]$. De manera que la continuidad nos garantiza la existencia de la raíz; sin embargo por simplicidad se supondrá que tal raíz es única, lo cual podría no ser cierto, como se ilustra en el siguiente ejemplo gráfico:



Procedimiento de Bisección

El procedimiento de Bisección consiste en dividir repetidamente a la mitad el subintervalo de [*a,b*] que contiene a la raíz y retener la mitad en donde *f* cambia de signo; para conservar a la raíz. Los pasos son los siguientes :

1. Evaluar que *f* sea continua en un intervalo [*a,b*] .

2. Evaluar si *f(a).f(b) < 0,* pues en caso contrario, no tendríamos garantizada la existencia de la raíz. Asumamos, que efectivamente la condición de signos contrarios de *f(a)* y *f(b)* se cumple.

3. Tomar $a_1 = a, b_1 = b$ y $p_1 = \frac{a_1 + b_1}{2}$; si $f(p_1) = 0$ entonces $p = p_1$ (es la raíz buscada), si no, entonces $f(p_1)$ tiene el mismo signo que $f(a_1)$ ó $f(b_1)$. Si $f(p_1)$ y $f(a_1)$ tienen el mismo signo, entonces la raíz esta en la otra mitad del subintervalo (derecho), es decir $f(p_1) \cdot f(b_1) < 0$.

4. Tomar $a_2 = p_1, b_2 = b_1$ y $p_2 = \frac{a_2 + b_2}{2}$; si $f(p_2) = 0$, entonces p_2 es la raíz buscada, en caso contrario, si $f(p_2)$ y $f(b_2)$ son del mismo signo, entonces la raíz está en (a_2, p_2). Se hace entonces:

5. $a_3 = a_2, b_3 = p_2$ y $p_3 = \frac{a_3 + b_3}{2}$. Si $f(p_3) = 0$, entonces p_3 es la raíz buscada. Si no, tomamos el subintervalo en donde esta la raíz, reasignamos los extremos del intervalo y continuamos con el proceso hasta aproximarnos a la raíz, con la precisión deseada, en una cantidad finita de pasos.

Como criterio para detener el proceso iterativo se considera lo siguiente. Tomar una tolerancia $\in > 0$, de la sucesión de puntos medios $p_1, p_2, \cdots, p_N$; hasta que una de las condiciones siguientes se satisface:

1. $|p_N - p_{N-1}| < \in$,

2. $\frac{|p_N - p_{N-1}|}{|p_N|} < \in$, $p_N \neq 0$,

3. $|f(p_N)| < \in$

4. Número máximo de iteraciones.

El primer y el segundo criterio están asociados al error absoluto y al error relativo respectivamente. De todos estos, el mejor criterio para detener el proceso esta dado por la expresión 2.; ya que en la 1., puede ocurrir que $P_N - P_{N-1}$ converja a Cero y la Sucesión $\{P_N\}$ divergir. También puede ocurrir que $f(P_N)$ este cercano a Cero y P_N diferir de P. Ver ejercicios propuestos 2.1.14 y 2.1.15 en Burden [1].

Consideraciones sobre el método de Bisección:

1. El procedimiento de Bisección, siempre converge, pero es de convergencia lenta.

2. El procedimiento de Bisección puede desechar una buena aproximación intermedia, a la solución o raíz.

3. El procedimiento de Bisección requiere de un intervalo pequeño en longitud, para garantizar una convergencia más rápida.

4. El procedimiento de Bisección se puede implementar con métodos más eficientes, utilizándole a él para dar una aproximación inicial a la raíz.

Algoritmo de Bisección

Para encontrar una solución de $f(x) = 0$ dada la función f continua en el intervalo $[a, b]$, y con $f(a)$ y $f(b)$ de signos diferentes.
ENTRADA: Extremos a, b; tolerancia TOL; máximo número de iteraciones N_o.

SALIDA: Solución aproximada p o mensaje de fracaso.

Paso 1 :Tomar $i = 1$;
$FA = f(a)$.
Paso 2 : Mientras que $i \le N_o$ seguir Pasos 3 – 6.
Paso 3 : Tomar $p = a + \dfrac{b-a}{2}$; (calcular la raíz)
$FP = f(p)$.
Paso 4 : Sí $FP = 0 \quad o \quad \dfrac{b-a}{2} < TOL$, entonces
SALIDA (p); (proceso exitoso)
PARAR.

Paso 5 : Tomar $i = i + 1$.

Paso 6 : Sí $FA \cdot FP > 0$ entonces tomar $a = p$;

$FA = FP$;

Si no tomar $b = p$.

Paso 7: **SALIDA** ('El método fracasó después de N_o iteraciones, N_o = ', N_o);

(Procedimiento completado sin éxito)

PARAR.

Programa de Bisección (Codificado en Turbo Pascal).

A continuación se presenta la codificación en el Lenguaje de Programación Turbo Pascal, del Algoritmo anterior, algo enriquecido con procedimientos de validación. La función a la cual se le va a encontrar la raíz en [0,1] es la función real continua sobre dicho intervalo, dada por $f(x) = \cos x - x^2$, se considerará como tolerancia o precisión de la raíz a 10^{-5}, y el número máximo de iteraciones será de 100, como método alternativo para no caer en un ciclo de iteraciones infinitas.

```
{************************************************************************}
{*    UNIVERSIDAD CENTROCCIDENTAL LISANDRO ALVARADO                     *}
{* DECANATO DE CIENCIAS Y TECNOLOGÍAS - DEPARTAMENTO DE MATEMATICAS     *}
{*  PROGRAMA DE LICENCIATURA EN CIENCIAS MATEMATICAS                    *}
{*               BARQUISIMETO                                           *}
{*                                                                      *}
{*      PROGRAMA PARA ENCONTAR UNA RAIZ REAL DE UNA FUNCION REAL        *}
{*      METODO : BISECCION                                              *}
{*      SOPORTE TEORICO : BURDEN-AYRES[1]: ANALISIS NUMERICO. 6ta. ed.  *}
{*                        pp. 48-52.                                    *}
{*      LENGUAJE : TURBO PASCAL                                         *}
{*      AUTOR : PROF. PEDRO E. ADAMES ROJAS                             *}
{*      E-MAIL : padames@delfos.ucla.edu.ve                             *}
{*      FECHA : ENERO, 2.001                                            *}
{*                                                                      *}
{************************************************************************}

PROGRAM BISECCION;
USES CRT;
CONST
     TOL = 1E-5;                        (* Tolerancia deseada           *)
     NMAX = 100;                        (* Numero Maximo de Iteraciones *)
VAR
   A,B : REAL;                          (* Extremos del Intervalo       *)
   FA,FB : REAL;                        (* F(A) y F(B)                  *)
   RAIZ, FRAIZ : REAL;                  (* Raiz Buscada e Imagen        *)
   CONTINUAR, INTERVALO : BOOLEAN;      (* Variables de Control         *)
   I : INTEGER;                         (* Contador de Iteraciones      *)
(* DECLARAR Y DEFINIR LA FUNCION *)
FUNCTION F (X : REAL) : REAL;           (* Definicion de la Funcion Real *)
BEGIN                                   (* a la que se le determinara la *)
   F := COS(X) - SQR(X)                 (* Raiz en el Intervalo dado     *)
END;
```

```
(* BLOQUE PRINCIPAL DEL PROGRAMA *)
BEGIN                                                          (****************)
     REPEAT                                                    (* Aca se da el *)
           CLRSCR;                                             (* Intervalo en *)
           WRITELN;                                            (* donde se de- *)
           WRITE(' EXTREMO IZQUIERDO DEL INTERVALO :  ');      (* sea la Raiz  *)
           READLN (A);                                         (* y se valida  *)
           WRITELN;                                            (* que la misma *)
           WRITE(' EXTREMO DERECHO DEL INTERVALO :  ');        (* exista en el *)
           READLN (B);                                         (****************)
           INTERVALO := A < B;
           FA := F(A);
           FB := F(B);
           CONTINUAR := FA*FB < 0;
     UNTIL CONTINUAR AND INTERVALO;
     CLRSCR;
     I := 0;
     WRITELN(' I         A            B          RAIZ        F(RAIZ)   ');
     WRITELN('---    ---------    ---------    ---------    ----------');
     REPEAT                                                          (* Calculo de las *)
           RAIZ := (A + B)/2;                                        (* Aproximaciones *)
           FRAIZ := F(RAIZ);                                         (* por el Metodo  *)
           WRITELN(I:3,' ',A:12:6,' ',B:12:6,' ',RAIZ:12:6,'  ',FRAIZ:12:6);
           IF FA*FRAIZ <= 0 THEN B := RAIZ                           (* de BISECCION   *)
              ELSE                                                   (* e Impresion de *)
                 BEGIN                                               (* los Resultados *)
                    A := RAIZ;                                       (* Parciales      *)
                    FA:= FRAIZ;                                      (****************)
                 END;
           IF (B - A) <= TOL THEN CONTINUAR := FALSE
              ELSE
                 I := SUCC(I);
     UNTIL ((NOT CONTINUAR) OR (I > NMAX));
(* BLOQUE DE IMPRESION DE RESULTADO FINAL *);
   IF I > NMAX THEN WRITELN(' EXCEDIDO EL NUMERO MAXIMO DE ITERACIONES ')
      ELSE
         BEGIN
             WRITELN;
             WRITE(' RAIZ APROXIMADA =  ',RAIZ:12:6);
             WRITE(' LOGRADA CON ',SUCC(I):3,' ITERACIONES');
         END;
   WRITELN;
   WRITE(' .... PULSE ENTER PARA TERMINAR ...');
   READLN
END.
```

Ejemplo 2.2.2.1. Usando el programa anterior para encontrar la raiz en [0,1] para la función real continua sobre dicho intervalo, dada por $f(x) = \cos x - x^2$, se obtiene la siguiente salida:

I	A	B	RAIZ	F(RAIZ)
0	0.000000	1.000000	0.500000	0.627583
1	0.500000	1.000000	0.750000	0.169189

2	0.750000	1.000000	0.875000	-0.124628
3	0.750000	0.875000	0.812500	0.027529
4	0.812500	0.875000	0.843750	-0.047248
5	0.812500	0.843750	0.828125	-0.009533
6	0.812500	0.828125	0.820313	0.009080
7	0.820313	0.828125	0.824219	-0.000206
8	0.820313	0.824219	0.822266	0.004442
9	0.822266	0.824219	0.823242	0.002119
10	0.823242	0.824219	0.823730	0.000957
11	0.823730	0.824219	0.823975	0.000376
12	0.823975	0.824219	0.824097	0.000085
13	0.824097	0.824219	0.824158	-0.000061
14	0.824097	0.824158	0.824127	0.000012
15	0.824127	0.824158	0.824142	-0.000024
16	0.824127	0.824142	0.824135	-0.000006

RAIZ APROXIMADA = 0.824135 LOGRADA CON 17 ITERACIONES
.... PULSE ENTER PARA TERMINAR ...

Ejemplos 2.2.2.2.Para encontrar una aproximación de $\sqrt[3]{25}$ correcta a 10^{-4}, usando el algoritmo de bisección, redefinimos el problema a encontrar un cero para $f(x) = x^3 - 25$, ya que con $f(x) = 0$, se obtiene la repuesta a la situación planteada. Considerando la tolerancia Tol = 10^{-4}, y usando el programa anterior en el intervalo [2,3] se obtiene:

I	A	B	RAIZ	F(RAIZ)
0	2.000000	3.000000	2.500000	-9.375000
1	2.500000	3.000000	2.750000	-4.203125
2	2.750000	3.000000	2.875000	-1.236328
3	2.875000	3.000000	2.937500	0.347412
4	2.875000	2.937500	2.906250	-0.452972
5	2.906250	2.937500	2.921875	-0.054920
6	2.921875	2.937500	2.929688	0.145710
7	2.921875	2.929688	2.925781	0.045261
8	2.921875	2.925781	2.923828	-0.004863
9	2.923828	2.925781	2.924805	0.020190
10	2.923828	2.924805	2.924316	0.007662
11	2.923828	2.924316	2.924072	0.001399
12	2.923828	2.924072	2.923950	-0.001732
13	2.923950	2.924072	2.924011	-0.000167

RAIZ APROXIMADA = 2.924011 LOGRADA CON 14 ITERACIONES
.... PULSE ENTER PARA TERMINAR ...

Teorema 2.2.2.1: Sea $f \in C\,[a,b]$ y supongamos que $f(a).\ f(b) < 0$. El procedimiento de bisección genera una sucesión $\{P_n\}$ que aproxima a P con la propiedad:

$\left|P_n - P\right| \le \dfrac{b-a}{2^n}$, n ≥ 1

Demostración:

$\forall n \ge 1:\ b_n - a_n = \dfrac{b-a}{2^{n-1}}$ y $P \in (a_n, b_n)$; como $P_n = \dfrac{a_n + b_n}{2}$

$$\left|P_n - P\right| \le \frac{1}{2}\left(b_n - a_n\right) = \frac{1}{2}\cdot\frac{b-a}{2^{n-1}} = \frac{b-a}{2^n} = \frac{b-a}{2^n} = 2^{-n}\left(b-a\right)$$

$$\therefore \left|P_n - P\right| \le \frac{b-a}{2^n}, \forall n \ge 1$$

$\therefore P_n \to P$ con una rapidez de convergencia $O(2^{-n})$

Este teorema no da cota para el error involucrado en la aproximación de P_n por P y es útil en el proceso de determinar el número de pasos máximo para garantizar la aproximación con la precisión deseada.

Ejemplo 2.2.2.3. En el ejercicio anterior, si deseamos calcular la cantidad de pasos o iteraciones necesarias para garantizar la precisión de 10^{-4}, de acuerdo al teorema previo, procedemos de la manera siguiente: $a = 2$, $b = 3$, la cota para el error es 10^{-4}, así que:

$\dfrac{3-2}{2^n} < 10^{-4} \Rightarrow 2^{-n} < 10^{-4} \Rightarrow n > \dfrac{-4\ln 10}{-\ln 2} \approx 13.2877 \quad \therefore n = 14.$ Lo cual coincide con el resultado encontrado.

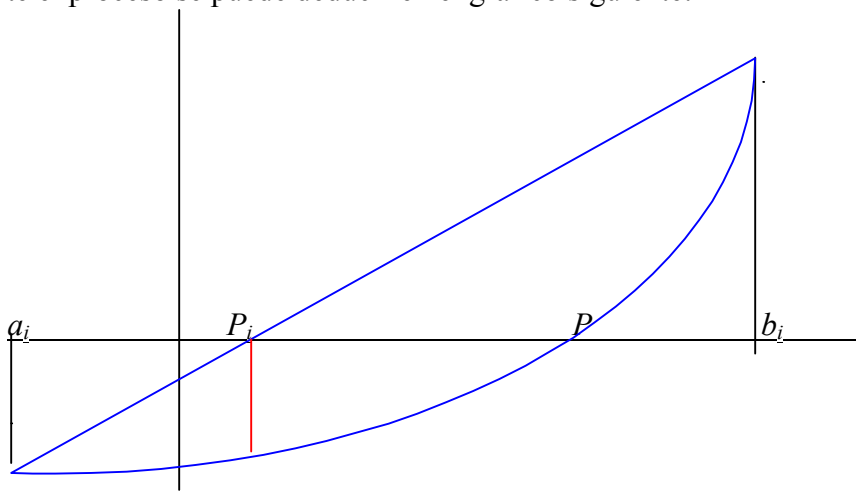
Como ya se comentó, el método de Bisección es lento en su convergencia, por lo que a continuación estudiaremos otro método que por lo general es más rápido y que se apoya tanto en intervalo como en el hecho de que para la función dada continua, ocurra cambio de signo, el es llamado Método de Regla Falsa o Posición Falsa.

2.2.3. METODO DE POSICIÓN FALSA-REGLA FALSA O REGULA FALSA

Sea f una función continua en $[a,b]$, dada por $f(x) = 0$; sea $\left[a_i, b_i\right] \subset \left[a,b\right]$, tal que este i-ésimo subintervalo contiene una raíz de $f(x) = 0$; el método de *Posición Falsa* consiste en lo siguiente: Calculamos la intersección con el eje X de la recta que une los puntos $\left(a_i, f(a_i)\right)$ y $\left(b_i, f(b_i)\right)$ denotamos por p_i a este punto de corte. Evaluamos:

Si $f(a_i).f(p_i) < 0$, definimos $a_{i+1} = a_i \wedge b_{i+1} = p_i$, en caso contrario definimos $a_{i+1} = p_i \wedge b_{i+1} = b_i$; y proseguimos de ésta manera encontrando subintervalos que contienen la raíz como en el caso de bisección, hasta llegar a tenerla con una tolerancia aceptable o a satisfacer un criterio de "detención", según nuestras especificaciones.

Gráficamente el proceso se puede deducir en el gráfico siguiente:



Deducción de la fórmula

La fórmula en que se apoya el Método de Regla Falsa se puede deducir del gráfico anterior y por semejanza de triángulos, al establecer:

$\frac{f(b_i) - f(a_i)}{b_i - a_i} = \frac{f(b_i)}{b_i - P_i}$, de donde: $b_i - P_i = \frac{f(b_i)}{f(b_i) - f(a_i)} \cdot (b_i - a_i)$, y despejando:

$$\boxed{P_i = b_i - \frac{f(b_i)(b_i - a_i)}{f(b_i) - f(a_i)}}$$

Algoritmo de Regla Falsa

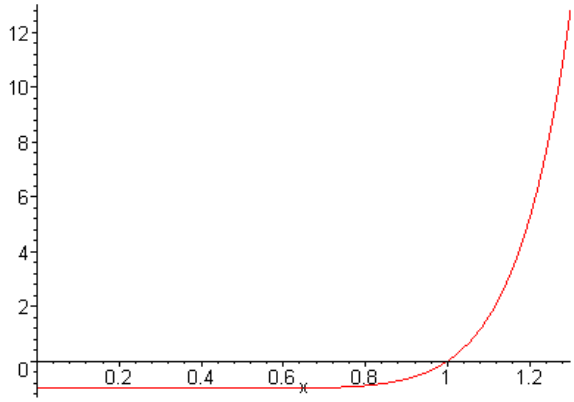
El algoritmo para Regla Falsa, funciona de manera similar en cuanto a criterio para asegurar la raíz al Método de Bisección, por lo que el mismo queda como sigue, al implementarle la fórmula anterior:

Objetivo: Encontrar una solución de $f(x) = 0$ dada la función f continua en el intervalo $[a, b]$, y con $f(a)$ y $f(b)$ de signos diferentes.
ENTRADA: Extremos a, b; tolerancia TOL; máximo número de iteraciones N_o.

SALIDA: Solución aproximada p o mensaje de fracaso.

Paso 1 :Tomar $i = 1$;
$FA = f(a)$.
Paso 2 : Mientras que $i \le N_o$ seguir Pasos 3 – 6.
Paso 3 : Tomar $p = b - \dfrac{f(b).(b-a)}{f(b)-f(a)}$; (calcular la raíz)
$FP = f(p)$.
Paso 4 : Sí $FP = 0$ o $\dfrac{b-a}{2} < TOL$, entonces
SALIDA (p); (proceso exitoso)
PARAR.
Paso 5 : Tomar $i = i + 1$.
Paso 6 : Sí $FA \,.\, FP > 0$ entonces tomar $a = p$;
$FA = FP$;
Si no tomar $b = p$.
Paso 7: **SALIDA** ('El método fracasó después de N_o iteraciones, N_o = ', N_o);
(Procedimiento completado sin éxito)
PARAR.

Es importante destacar que el Método de Regla Falsa, por lo general es de una mayor rapidez en su convergencia, comparado con el Método de Bisección. Sin embargo un caso de convergencia lenta del método de Regla Falsa, se presenta con: $f(x) = x^{10} - 1$, en $a = 0$, $b = 1.3$, como se puede observar en la gráfica siguiente.



Programa de Regla Falsa (Codificado en Turbo Pascal).

El programa que se muestra a continuación, codificado en Turbo PASCAl, calcula por el método de Regla Falsa, la raíz de la función $f(x)= cos(x) – x^2$

```
{************************************************************************}
{*    UNIVERSIDAD CENTROCCIDENTAL LISANDRO ALVARADO                     *}
{* DECANATO DE CIENCIAS Y TECNOLOGÍA - DEPARTAMENTO DE MATEMATICAS      *}
{*  PROGRAMA DE LICENCIATURA EN CIENCIAS MATEMATICAS                    *}
{*               BARQUISIMETO                                           *}
{*                                                                      *}
{*      PROGRAMA PARA ENCONTAR UNA RAIZ REAL DE UNA FUNCION REAL        *}
{*      METODO : REGLA FALSA O INTERPOLACION LINEAL                     *}
{*      SOPORTE TEORICO : CURTIS, GERALD : ANALISIS NUMERICO. 2da.  ed. *}
{*                        pp. 8 -10.                                    *}
{*      LENGUAJE : TURBO PASCAL                                         *}
{*      AUTOR : PROF. PEDRO E. ADAMES ROJAS                             *}
{*      E-MAIL : padames@delfos.ucla.edu.ve                             *}
{*      FECHA : ENERO, 2.001                                            *}
{*                                                                      *}
{************************************************************************}

PROGRAM REGLA_FALSA;
USES CRT;
CONST
     TOL = 1E-5;                          (* Tolerancia deseada           *)
     NMAX = 100;                          (* Numero Maximo de Iteraciones  *)
VAR
   A,B : REAL;                            (* Extremos del Intervalo       *)
   FA,FB : REAL;                          (* F(A) y F(B)                  *)
   RAIZ, FRAIZ : REAL;                    (* Raiz Buscada e Imagen        *)
   CONTINUAR, INTERVALO : BOOLEAN;        (* Variables de Control         *)
   I : INTEGER;                           (* Contador de Iteraciones      *)
(* DECLARAR Y DEFINIR LA FUNCION *)
FUNCTION F (X : REAL) : REAL;             (* Definicion de la Funcion Real *)
BEGIN                                     (* a la que se le determinara la *)
   F := COS(X) - SQR(X)                   (* Raiz en el Intervalo dado     *)
END;
(* BLOQUE PRINCIPAL DEL PROGRAMA *)
BEGIN                                                        (***************)
     REPEAT                                                  (* Aca se da el *)
           CLRSCR;                                           (* Intervalo en *)
           WRITELN;                                          (* donde se de- *)
           WRITE(' EXTREMO IZQUIERDO DEL INTERVALO :  ');    (* sea la Raiz  *)
           READLN (A);                                       (* y se valida  *)
           WRITELN;                                          (* que la misma *)
           WRITE(' EXTREMO DERECHO DEL INTERVALO :  ');      (* exista en el *)
           READLN (B);                                       (***************)
           INTERVALO := A < B;
           FA := F(A);
           FB := F(B);
           CONTINUAR := FA*FB < 0;
     UNTIL CONTINUAR AND INTERVALO;
     CLRSCR;
     I := 0;
     WRITELN(' I        A           B          RAIZ          F(RAIZ)   ');
```

```
     WRITELN('---    ----------  ---------- ------------   -------------');
     REPEAT                                              (* Calculo de las *)
           RAIZ := B - FB*(B - A)/(FB - FA);             (* Aproximaciones *)
           FRAIZ := F(RAIZ);                             (* por el Metodo  *)
           WRITELN(I:3,' ',A:12:9,' ',B:12:9,' ',RAIZ:12:9,'  ',FRAIZ:12:9);
           IF FA*FRAIZ <= 0 THEN B := RAIZ               (* de Regla Falsa *)
              ELSE                                       (* e Impresion de *)
                 BEGIN                                   (* los Resultados *)
                    A := RAIZ;                           (* Parciales      *)
                    FA:= FRAIZ;                          (******************)
                 END;
           IF (B - A) <= TOL THEN CONTINUAR := FALSE
              ELSE
                 I := SUCC(I);
     UNTIL ((NOT CONTINUAR) OR (I > NMAX));
(* BLOQUE DE IMPRESION DE RESULTADO FINAL *);
   IF I > NMAX THEN WRITELN(' EXCEDIDO EL NUMERO MAXIMO DE ITERACIONES ')
      ELSE
         BEGIN
             WRITELN;
             WRITE(' RAIZ APROXIMADA =  ',RAIZ:12:9);
             WRITE(' LOGRADA CON ',SUCC(I):3,' ITERACIONES');
         END;
   WRITELN;
   WRITE(' .... PULSE ENTER PARA TERMINAR ...');
   READLN
END.
```

Produciendo los siguientes resultados:

I	A	B	RAIZ	F(RAIZ)
0	0.000000000	1.000000000	0.685073357	0.305047129
1	0.685073357	1.000000000	0.810693650	0.031771672
2	0.810693650	1.000000000	0.822931603	0.002858426
3	0.822931603	1.000000000	0.824025820	0.000253672
4	0.824025820	1.000000000	0.824122874	0.000022485
5	0.824122874	1.000000000	0.824131476	0.000001993
6	0.824131476	1.000000000	0.824132238	0.000000177
7	0.824132238	1.000000000	0.824132306	0.000000016
8	0.824132306	1.000000000	0.824132312	0.000000001
9	0.824132312	1.000000000	0.824132312	0.000000000
10	0.824132312	1.000000000	0.824132312	0.000000000
11	0.824132312	1.000000000	0.824132312	0.000000000
12	0.824132312	1.000000000	0.824132312	-0.000000000

RAIZ APROXIMADA = 0.824132312 LOGRADA CON 13 ITERACIONES
.... PULSE ENTER PARA TERMINAR ...

Ejemplo 2.2.3.1. En la ejecución del programa anterior observamos que la raíz fue lograda en 13 iteraciones, mientras que este mismo caso para el método de Bisección, la raíz se obtuvo en 16 iteraciones. Para el caso de $f(x) = x^3 - 25$, los resultados siguientes :

I	A	B	RAIZ	F(RAIZ)
0	2.000000	3.000000	2.894737	-0.743549
1	2.894737	3.000000	2.923265	-0.019304
2	2.923265	3.000000	2.923999	-0.000493
3	2.923999	3.000000	2.924017	-0.000013
4	2.924017	3.000000	2.924018	-0.000000
5	2.924018	3.000000	2.924018	-0.000000
6	2.924018	3.000000	2.924018	-0.000000
7	2.924018	3.000000	2.924018	0.000000

RAIZ APROXIMADA = 2.924018 LOGRADA CON 8 ITERACIONES
.... PULSE ENTER PARA TERMINAR ...

De manera que la observación sobre la comparación de la rapidez de éste método contra el de Bisección queda puesta en evidencia. Mientras que para la función $f(x) = x^{10} - 1$, con una tolerancia de 10^{-4}, y un número máximo de 100 iteraciones, observamos que este se alcanza sin lograr encontrar la raíz, se deja como ejercicio verificar esta afirmación.

2.2.4. FUNCIONES DE PUNTO FIJO

Un punto fijo para una función g, es un número p para el cual $g(p) = p$. El problema de encontrar raíces y el problema de punto fijo son equivalente, en el sentido siguiente: Dado el problema de encontrar la raíz $f(p) = 0$; es equivalente a si definimos $g(x) = x - f(x)$, en donde $g(p) = p - f(p)$, o sea $g(p) = p$. Recíprocamente si g tiene un punto fijo en p ($g(p)=p$), entonces $f(x)=x - g(x)$, tiene un cero en p.

Ejemplo 2.2.4.1:
$g(x) = x^2 - 2, con - 2 \le x \le 3,$ tiene en 2 y en –1 puntos fijos; por cuanto:
$g(-1) = 1 - 2 = -1 \wedge g(2) = 4 - 2 = 2$

Aunque nuestro objetivo es encontrar raíces, el manejo de punto fijo, puede proveernos en algunos casos de una técnica más poderosa para encontrar raíces. El problema ahora es entonces de cuando saber la existencia de punto fijo, lo cual nos lo provee el siguiente teorema.

Teorema 2.2.4.1:
i) Si $g \in C[a,b]$ y $g(x) \in [a,b], \forall x \in [a,b]$, entonces g tiene un punto fijo en $[a,b]$.

ii) Si adicionalmente, *g´(x)* existe en *(a,b) y K* es una constante positiva menor que 1, con $|g'(x)| \le K < 1,\ \forall x \in [a,b]$, entonces *g* tiene un punto fijo en [*a,b*] es único.

Demostración:

i) Existencia

Si *g(a)= a o g(b)= b,* entonces *a o b* serían puntos fijos.

Supongamos que no es el caso de esta trivialidad. Como $g(x) \in [a,b]$ y $a < b$, $g(a) > a \wedge g(b) < b$. Definamos $h(x) = g(x) - x$, *h* es continua en [*a,b*] por ser lo *g(x)* y $x \cdot h(a) = g(a) - a > 0 \wedge h(b) = g(b) - b < 0$; luego por el Teorema del Valor Intermedio, $\exists\ p \in (a,b)$ tal que $h(p) = 0 \Rightarrow g(p) - p = 0$, o sea *g(p)= p*; por lo que *p* sería punto fijo.

ii) Unicidad

Supongamos que *p y q* son dos puntos fijos en [*a,b*] y que *p≠q*. Luego por el Teorema del Valor Medio, $\exists\ \xi \in (p,q) \ni \dfrac{g(q) - g(p)}{q - p} = g'(\xi)$, de donde $g(q) - g(p) = g'(\xi)(q - p)$. Por otra parte:

$$|q - p| = |g(q) - g(p)| = |g'(\xi)(q - p)| = |g'(\xi)| \cdot |q - p|; |g'(\xi)| < 1$$

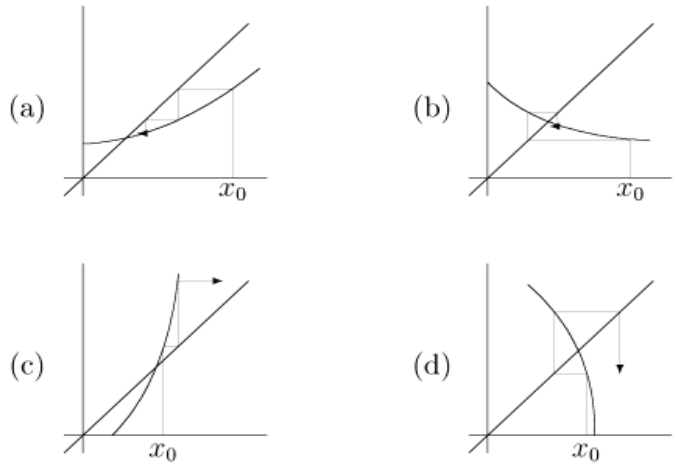
$$< 1 \cdot |q - p| = |q - p|$$; o sea que:

$|q - p| < |q - p|$ y esto es una contradicción que proviene de suponer que $p \neq q$; por lo tanto el punto fijo es único.

Proceso Iterativo del Punto Fijo o Iteración Funcional

Para resolver *g(x)= x*, elegimos una aproximación inicial P_o y generamos una sucesión $\{P_n\}_{n=0}^{\infty}$ haciendo $P_n = g(P_{n-1})$ para n ≥ 1. Si la sucesión $\{P_n\}$ converge a *P* y *g* es continua entonces: $p = \lim_{n\to\infty} P_n = \lim_{n\to\infty} g(P_{n-1}) = g(\lim_{n\to\infty} P_{n-1}) = g(p)$; y así se obtendrá la solución a $g(x) = x$

Ilustraciones



Situaciones de Convergencia al Punto Fijo

Algoritmo de Punto Fijo

Para encontrar una solución de *P = g(P),* dada una aproximación inicial P_0.

ENTRADA: Aproximación inicial P_o, Tolerancia TOL, Número máximo de iteraciones N_0.

SALIDA: Solución aproximada *P* o mensaje de fracaso.

PASO 1: Tomar i = 1.

PASO 2: Mientras i $\leq N_0$, haces **PASOS 3-6.**

PASO 3: Tomar $P = g(P_0)$. (Calculo de P_i)

PASO 4: Si $|P - P_0| <$ TOL, entonces

SALIDA (P); (Proceso exitoso)

PAPAR.

PASO 5: Tomar i = i + 1.

PASO 6: Tomar $P_0 = P$. (Redefinir P_o).

PASO 7 : SALIDA ('El método fracaso después de N_0 iteraciones, $N_0 =$ ', N_0); (Proceso no exitoso)

PARAR.

Programa en Turbo Pascal para la Iteración al Punto Fijo

A continuación se presenta la codificación del algoritmo anterior, incluyendo procesos de validación, para la función $g(x) = \sqrt{\frac{10}{x+4}}$; usando como punto de arranque o inicio $x_0 = 0$, tolerancia 10^{-9}, y un número máximo de 100 iteraciones.

```
{************************************************************************}
{*    UNIVERSIDAD CENTROCCIDENTAL LISANDRO ALVARADO                     *}
{* DECANATO DE CIENCIAS Y TECNOLOGÍA - DEPARTAMENTO DE MATEMATICAS      *}
{*  PROGRAMA DE LICENCIATURA EN CIENCIAS MATEMATICAS                    *}
{*               BARQUISIMETO                                           *}
{*                                                                      *}
{*      PROGRAMA PARA ENCONTAR UNA RAIZ REAL DE UNA FUNCION REAL        *}
{*      METODO : ITERACCION AL PUNTO FIJO                               *}
{*      SOPORTE TEORICO : BURDEN & AYRES : ANALISIS NUMERICO. 6ta.  ed. *}
{*                        pp. 54-63.                                    *}
{*      LENGUAJE : TURBO PASCAL                                         *}
{*      AUTOR : PROF. PEDRO E. ADAMES ROJAS                             *}
{*      E-MAIL : padames@delfos.ucla.edu.ve                             *}
{*      FECHA : ENERO, 2.001                                            *}
{*                                                                      *}
{************************************************************************}

PROGRAM ITERACION_PTP_FIJO;
USES CRT;
CONST
     TOL  = 1E-9;
     NMAX = 100;
VAR
   G1: REAL;
   X0: REAL;
   RAIZ,ERROR:BOOLEAN;
   I: INTEGER;
(*********************************************************************)
(*   DECLARAR  Y  DEFINIR  LA  FUNCION                               *)
(*********************************************************************)
 FUNCTION G(X:REAL):REAL;
 BEGIN
   G:= SQRT(10/(X + 4));
 END;
(*********************************************************************)
(*  BLOQUE  PRINCIPAL  DEL  PROGRAMA                                 *)
(*********************************************************************)
 BEGIN
    I:= 1;
    RAIZ:=FALSE;
    ERROR:=FALSE;
       CLRSCR;
       WRITELN;
       WRITE (' PUNTO DE INICIO :');
       READLN(X0);
    CLRSCR;
    WRITELN(' I       X0           G(X0)     ');
    WRITELN('---  -----------   ------------ ');
    (*********************************************************************)
    (* Desarrollo del Metodo de Iteraccion al Punto Fijo para Raices de  *)
    (* Ecuaciones no lineales, dada una aproximacion inicial a dicha raiz *)
    (*********************************************************************)
    WHILE ((I<=NMAX) AND (NOT RAIZ) AND (NOT ERROR)) DO
       BEGIN
        G1:= G(X0);
        BEGIN
```

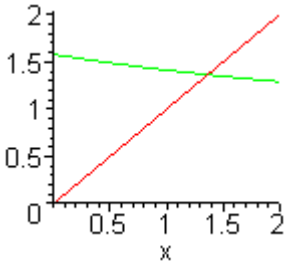
```
            WRITELN(I:2,'   ',X0:11:9,'   ',G1:11:9,'   ','   ');
            IF ((ABS(G1)<=TOL) OR (ABS(G1-X0)<=TOL)) THEN
                BEGIN
                   RAIZ := TRUE;
                END
            ELSE
               BEGIN
                  I := I + 1;
                  X0:= G1;
               END
         END;
        END;
    IF I > NMAX THEN WRITELN(' EXCEDIDO EL NUMERO MAXIMO DE ITERACIONES ')
       ELSE
          BEGIN
             WRITELN;
             WRITE(' RAIZ APROXIMADA = ', G1:12:8);
             WRITE(' LOGRADA CON ',I:2,' ITERACIONES');
          END;
    WRITELN;
    WRITE('....PULSE <ENTER> PARA TERMINAR ....');
    READLN
END.
```

Ejemplo 2.2.4.1 : La ejecución del programa anterior para la función $g(x)=\sqrt{\frac{10}{x+4}}$; usando como punto de arranque o inicio x_0 = 0, produce el siguiente resultado, con impresión de los pasos intermedios:

I	X0	G(X0)
1	0.000000000	1.581138830
2	1.581138830	1.338562294
3	1.338562294	1.368635632
4	1.368635632	1.364796925
5	1.364796925	1.365285118
6	1.365285118	1.365223002
7	1.365223002	1.365230905
8	1.365230905	1.365229900
9	1.365229900	1.365230028
10	1.365230028	1.365230012
11	1.365230012	1.365230014
12	1.365230014	1.365230013

RAIZ APROXIMADA = 1.36523001
LOGRADA CON 12 ITERACIONES
....PULSE <ENTER> PARA TERMINAR ....

La representación gráfica del problema es :



TEOREMA DE PUNTO FIJO (2.2.4.2): Sea $g \in C[a,b]$ y $g(x) \in [a,b], \forall x \in [a,b]$, entonces g tiene un punto fijo en $[a,b]$. Si además, g´(*x*) existe en (*a,b*) y $|g'(x)| \le K < 1;\ \forall x \in (a,b)$ y K constante positiva. Si P_0 es cualquier número en [*a,b*], entonces la sucesión definida por $P_n = g(P_{n-1}), n \ge 1$; converge al único punto fijo *P* en [*a,b*].

<u>*Demostración*</u>:

Por el teorema anterior la existencia del único punto fijo *P* en [*a,b*] está garantizada. Como *g(x)* $\in$ [*a,b*], $\forall P_n \in [a,b]$; para cualquier *n*, *g(P_n)* $\in$ [*a,b*] y por el Teorema del Valor Medio:

$$|P_n - P| = |g(P_{n-1}) - g(P)| = |g'(\xi)(P_{n-1} - P)| = |g'(\xi)||P_{n-1} - P|$$

$$\le K|P_{n-1} - P| \text{ ya que } \xi \in (a,b). \text{ Aplicando inductivamente el mismo criterio}$$

$|P_n - P| \le K^2|P_{n-2} - P| \le \cdots \le K^n |P_0 - P|$ y como $K < 1 \Rightarrow lim_{n\to\infty} K^n = 0$; por lo tanto:

$lim_{n\to\infty}|P_n - P| \le lim_{n\to\infty} K^n|P_0 - P| = 0 \Rightarrow \{P_n\}_{n=0}^{\infty} \to P$

COROLARIO 2.2.4.1: Si *g* satisface las hipótesis del teorema anterior, una cota para el error involucrado al usar P_n para aproximar a *P* está dado por:

$$|P_n - P| \le K^n \cdot max\{P_0 - a, b - P_0\}, \forall n \ge 1$$

<u>*Demostración*</u> *:*

$|P_n - P| \le K^n \cdot |P_0 - P| \le K^n .max\{P_0 - a, b - P_0\}, \forall n \ge 1$

COROLARIO 2.2.4.2: Si *g* satisface las hipótesis del teorema anterior entonces:

$$|P_n - P| \le \frac{K^n}{1-K}|P_1 - P_0|, \forall n \ge 1$$

<u>*Demostración*</u>*:*

Para $n \ge 1$ se tiene: $|P_{n+1} - P_n| = |g(P_n) - g(P_{n-1})| \le K|P_n - P_{n-1}| \le \cdots \le K^n|P_1 - P_0|$ y para $m > n \ge 1$ se cumple que :

$$\begin{aligned}|P_m - P_n| &= |P_m - P_{m-1} + P_{m-1} - P_{m-2} + P_{m-2} - \cdots + P_{n+1} - P_n| \\ &\le |P_m - P_{m-1}| + |P_{m-1} - P_{m-2}| + \cdots + |P_{n+1} - P_n| \\ &\le K^{m-1}|P_1 - P_0| + K^{m-2}|P_1 - P_0| + \cdots + K^n|P_1 - P_0| \\ &= K^n\left(1 + K + K^2 + \cdots + K^{m-n-1}\right)|P_1 - P_0|\end{aligned}$$

como $lim_{m\to\infty} P_m = P$, entonces:

$$|P - P_n| = lim_{m\to\infty}|P_m - P_n| \le K^n|P_1 - P_0| \cdot \sum_{i=0}^{\infty} K^i = K^n \cdot \frac{1}{1-K} \cdot |P_1 - P_0|$$

(serie geométrica $0 < K < 1$ converge a $\frac{1}{1-K}$)

de donde $\cdot|P_n - P| \le \frac{K^n}{1-K} \cdot |P_1 - P_0|$.

Algunas consideraciones, sobre los corolarios:

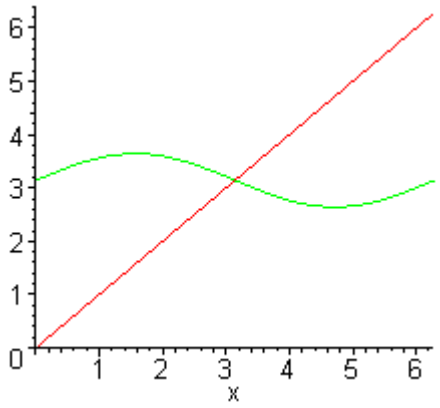
1. Los corolarios anteriores relacionan la rapidez de la convergencia del método de punto fijo; mientras más pequeño sea *K*, más rápida será la convergencia.

2. Si *K* está cerca de 1, la convergencia puede ser lenta.

Ejemplo 2.2.4.2.: Utilizando el Teorema del Punto Fijo, mostremos que la función $g(x) = \pi + \frac{1}{2} senx$, tiene un único punto fijo en [0,2π], y mediante iteración al punto fijo, encontraremos una aproximación con una precisión de 10^{-2} la cual compararemos con la estimación teórica sobre el número de iteraciones que se requieren para garantizar tal precisión.

Veamos: Evidentemente que la función $g(x) = \pi + \frac{1}{2} senx$ es continua en [0,2π], por se producto y suma de funciones continuas en R, además es diferenciable en (0,2π), y su derivada esta dada por $g'(x) = \frac{1}{2}\cos x$ la cual también es continua en todo R y en particular en cualquier subintervalo de R. Por otra parte como en el intervalo dado, $g'(x) = 0$ sólo en $x = \frac{\pi}{2}$ o $x = \frac{3\pi}{2}$, aplicando el Teorema del Valor Extremo, se tiene: $g(0) = \pi$, $g(\pi/2) = \pi + 0{,}5$; $g(3\pi/2) = \pi - 0{,}5$ y $g(2\pi) = \pi$, por lo tanto $\pi - \frac{1}{2} \le g(x) \le \pi + \frac{1}{2}, \forall x \in [0,2\pi]$, por lo que $g(x) \in [0,2\pi]$; así que la existencia de punto fijo para *g* está garantizada. Por otra parte, como $|g'(x)| = \left|\frac{1}{2}\cos x\right| \le \frac{1}{2} = 0{,}5 < 1$. Tomando $k = 0{,}5$ se garantiza por teorema previo la unicidad del punto fijo y la convergencia al punto fijo de la función *g*. Como $g(p) = p$, se deduce que $p = \pi$. Tomando como punto de arranque $p_0 = 0$, $g(p_0) = \pi = p_1$ que es el punto fijo, de manera que se ha alcanzado con 2 iteraciones. Si hacemos la estimación teórica por los corolarios enunciados, se tiene: $|p_n - p| \le \frac{(0{,}5)^n}{1-0{,}5}|0 - \pi| = \frac{1}{2^{n-1}}\pi < 10^{-2}$ de donde resolviendo se tiene que $n > 9{,}29$; por lo que con al menos $n = 10$ se garantiza la precisión deseada, esto es cierto porque lo logramos a penas con 2 iteraciones.

La representación gráfica de la situación planteada se muestra a continuación:



Ejemplo 2.2.4.3: Refiere Scheid [2], en el año 1225 Leonardo de Pisa estudio la ecuación: $f(x) = x^3 + 2x^2 + 10x - 20 = 0$, y produjo la raíz x = 1,3668808107. Nadie sabe por qué método hallo Leonardo este valor pero el resultado es notable para su época. Utilizando el método de iteración al punto fijo de puede obtener el resultado obtenido por Leonardo, veamos: Considerando $x = \dfrac{20}{x^2 + 2x + 10} = g(x)$ y tomado a 1 como punto de inicio, se genera la tabla siguiente:

I	X0	G(X0)	I	X0	G(X0)
1	1.0000000000	1.5384615385	2	1.5384615385	1.2950191571
3	1.2950191571	1.4018253094	4	1.4018253094	1.3542093904
5	1.3542093904	1.3752980925	6	1.3752980925	1.3659297882
7	1.3659297882	1.3700860034	8	1.3700860034	1.3682410236
9	1.3682410236	1.3690598120	10	1.3690598120	1.3686963976
11	1.3686963976	1.3688576886	12	1.3688576886	1.3687861026
13	1.3687861026	1.3688178744	14	1.3688178744	1.3688037731
15	1.3688037731	1.3688100317	16	1.3688100317	1.3688072540
17	1.3688072540	1.3688084868	18	1.3688084868	1.3688079396
19	1.3688079396	1.3688081825	20	1.3688081825	1.3688080747
21	1.3688080747	1.3688081225	22	1.3688081225	1.3688081013
23	1.3688081013	1.3688081107	24	1.3688081107	1.3688081065
25	1.3688081065	1.3688081084	26	1.3688081084	1.3688081076

Así que en la iteración 26 obtenemos la aproximación lograda por Leonardo.

Ejemplo 2.2.4.4 : Analicemos por qué es lenta la convergencia del ejemplo anterior. Como $\dfrac{20}{x^2 + 2x + 10} = g(x)$, entonces si consideremos a x en el intervalo [1,2], se puede verificar que

$\frac{20}{18} \le g(x) \le \frac{20}{13}$, de donde $g(x) \in [1,2]$. Adicionalmente, como *g* es diferenciable en dicho intervalo y se puede verificar que en este caso $\frac{120}{18^2} \le g'(x) \le \frac{120}{169}$, por lo que $|g'(x)| \le \frac{120}{169} \approx 0.71 = k < 1$. Esto garantiza la convergencia del método hacia el único punto fijo, pero como *k* relativamente está cerca de 1, dicha convergencia será lenta.

2.2.5. METODO DE NEWTON-RAPHSON

Para derivar éste método, se considera el problema de encontrar las raíces o los ceros de *f(x)=0*, para ello se utilizará el polinomio de Taylor, y asumiremos que *f* es una función continuamente diferenciable 2 veces en el intervalo [*a,b*]; o sea $f \in C^2[a,b]$. Sea $\bar{x} \in [a,b]$; una aproximación a *p* (raíz de *f*) tal que *f'(* $\bar{x}$ *)≠ 0* y $|\bar{x} - p|$ es "pequeño". Mediante el polinomio de Taylor de 1er grado para *f(x)* alrededor de $\bar{x}$

$$f(x) = f(\bar{x}) + (x - \bar{x}) f'(\bar{x}) + \frac{(x - \bar{x})^2}{2} \cdot f''(\xi(x))$$

en donde $\xi(x)$ está entre *x* y $\bar{x}$. Como *f(p) = 0*, con *x = p*:

$0 = f(\bar{x}) + (p - \bar{x}) f'(\bar{x}) + \frac{(p - \bar{x})^2}{2} \cdot f''(\xi(p))$; descartando el término del error:

$$0 \approx f(\bar{x}) + (p - \bar{x}) f'(\bar{x})$$

$$-\frac{f(\bar{x})}{f'(\bar{x})} + \bar{x} \approx p$$

$$p \approx \bar{x} - \frac{f(\bar{x})}{f'(\bar{x})}; n \ge 1$$

lo cual debe ser una mejor aproximación a *p* que $\bar{x}$

El método de Newton-Raphson involucra el generar una sucesión $\{P_n\}$ definida por

$$P_n = P_{n-1} - \frac{f(P_{n-1})}{f'(P_{n-1})}; n \ge 1$$

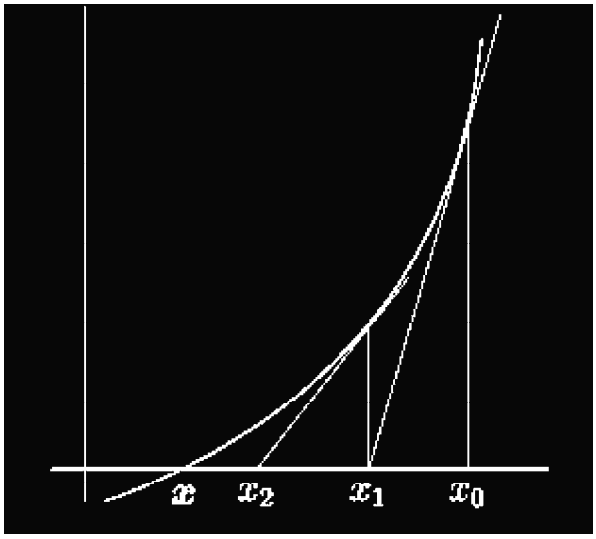
La fórmula anterior también puede derivarse gráficamente, al apoyarse en semejanza de triángulos, así tenemos:

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

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

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

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



Esquema gráfico del Método de Newton-Raphson

Algoritmo de Newton-Raphson

Objetivo: Para encontrar una solución de *f(x) = 0* dada una aproximación inicial P_o.
ENTRADA: Aproximación inicial P_o. Tolerancia *TOL*; Número máximo de iteraciones N_o.
SALIDA: Solución aproximada *P* o mensaje de fracaso.
Paso 1: Tomar $i = 1$.
Paso 2: Mientras que $i \leq No$, seguir **Pasos 3-6**.

Paso 3: Tomar $P = P_0 - \frac{f(P_0)}{f'(P_0)} \cdot$(Calculo de P_i).

Paso 4: Si $|P - P_0| < TOL$ entonces

SALIDA (*P*); (Proceso exitoso)
PARAR.
Paso 5: Tomar $i = i+1$.

Paso 6: Tomar $P_o = P$. (redefinición P_o).
Paso 7: SALIDA (´Él método fracaso después de ´,*No*,´íteraciones´);
PARAR.

Programa codificado en Turbo Pascal para el Método de Newton-Raphson

Para ilustrar la ejecución de un programa computarizado codificado en el Lenguaje de Programación Turbo Pascal, utilizado el Método de Newton-Raphson, se considerará que deseamos determinar una raíz de la función: $f(x) = 4\cos(x) - e^x$, con una precisión de 10^{-9} y un máximo de 100 iteraciones como criterio para parar el proceso. En estas condiciones el programa fuente podría ser el siguiente:

```
{************************************************************************}
{*    UNIVERSIDAD CENTROCCIDENTAL LISANDRO ALVARADO                     *}
{* DECANATO DE CIENCIAS Y TECNOLOGÍA - DEPARTAMENTO DE MATEMATICAS      *}
{*  PROGRAMA DE LICENCIATURA EN CIENCIAS MATEMATICAS                    *}
{*                BARQUISIMETO                                          *}
{*                                                                      *}
{*      PROGRAMA PARA ENCONTAR UNA RAIZ REAL DE UNA FUNCION REAL        *}
{*      METODO : NEWTON RAPSON                                          *}
{*      SOPORTE TEORICO : BURDEN & AYRES : ANALISIS NUMERICO. 6ta.  ed. *}
{*                        pp. 65-69.                                    *}
{*      LENGUAJE : TURBO PASCAL                                         *}
{*      AUTOR : PROF. PEDRO E. ADAMES ROJAS                             *}
{*      E-MAIL : padames@delfos.ucla.edu.ve                             *}
{*      FECHA : ENERO, 2001                                             *}
{*                                                                      *}
{************************************************************************}

PROGRAM NEWTON_RAPHSON;
USES CRT;
CONST
     TOL  = 1E-9;
     NMAX = 100;
VAR
   F1,FP: REAL;
   X0,X1:REAL;
   RAIZ,ERROR:BOOLEAN;
   I: INTEGER;
(*********************************************************************)
(*   DECLARAR  Y  DEFINIR  LA  FUNCION                               *)
(*********************************************************************)
 FUNCTION F(X:REAL):REAL;
 BEGIN
   F:= 4*cos(x) - exp(X);
 END;
 (*********************************************************************)
 (*   DECLARAR  Y  DEFINIR  LA  DERIVADA  DE  LA  FUNCION            *)
 (*********************************************************************)
 FUNCTION FPRIMA(Y:REAL):REAL;
 BEGIN
  FPRIMA:= -4*SIN(Y) -EXP(Y);
```

```
 END;
(*********************************************************************)
(*  BLOQUE  PRINCIPAL  DEL  PROGRAMA                                 *)
(*********************************************************************)
 BEGIN
    I:= 1;
    RAIZ:=FALSE;
    ERROR:=FALSE;
       CLRSCR;
       WRITELN;
       WRITE (' PUNTO DE INICIO :');
       READLN(X0);
    CLRSCR;
    WRITELN(' I       X1          F(X1)         F''(X1)        FP        ');
    WRITELN('---  -----------   ------------   -----------   ----------- ');
    (*********************************************************************)
    (* Desarrollo del Metodo de Newton-Raphson para localizar Raices de   *)
    (* Ecuaciones no lineales, dada una aproximacion inicial a dicha raiz *)
    (*********************************************************************)
    WHILE ((I<=NMAX) AND (NOT RAIZ) AND (NOT ERROR)) DO
       BEGIN
        F1:= F(X0);
        FP:= FPRIMA(X0);
        IF FP = 0 THEN
          BEGIN
             ERROR := TRUE;
             WRITELN (' LA DERIVADA SE ANULA, NO APLICABLE EL METODO ');
          END
         ELSE
          BEGIN
           X1 := X0 -F1/FP;
           WRITELN(I:2,'   ',X0:11:9,'   ',F1:11:9,'   ',FP:11:9,'
',X1:11:9,' ');
           IF ((ABS(F1)<=TOL) OR (ABS(X1-X0)<=ABS(X1)*TOL)) THEN
               BEGIN
                  RAIZ := TRUE;
               END
           ELSE
              BEGIN
                 I := I + 1;
                 X0:= X1;
              END
          END;
        END;
    IF I > NMAX THEN WRITELN(' EXCEDIDO EL NUMERO MAXIMO DE ITERACIONES ')
       ELSE
          BEGIN
             WRITELN;
             WRITE(' RAIZ APROXIMADA = ', X1:12:8);
             WRITE(' LOGRADA CON ',I:2,' ITERACIONES');
          END;
    WRITELN;
    WRITE('....PULSE <ENTER> PARA TERMINAR ....');
    READLN
 END.
```

Ejemplo 2.2.5.1 : El programa anterior ejecutado para la función $f(x) = 4 \cos x - e^x$, produce el siguiente resultado:

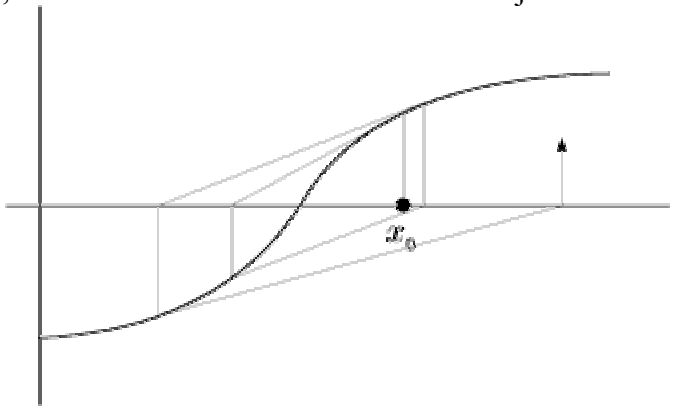
I	X1	F(X1)	F'(X1)	FP
1	0.000000000	3.000000000	-1.000000000	3.000000000
2	3.000000000	-24.045506909	-20.650016955	1.835569629
3	1.835569629	-7.315465870	-10.129311364	1.113362007
4	1.113362007	-1.277986440	-6.633331400	0.920700678
5	0.920700678	-0.089999024	-5.695152851	0.904897938
6	0.904897938	-0.000616281	-5.617128079	0.904788223
7	0.904788223	-0.000000030	-5.616585784	0.904788218
8	0.904788218	0.000000000	-5.616585758	0.904788218

RAIZ APROXIMADA = 0.90478822 LOGRADA CON 8 ITERACIONES

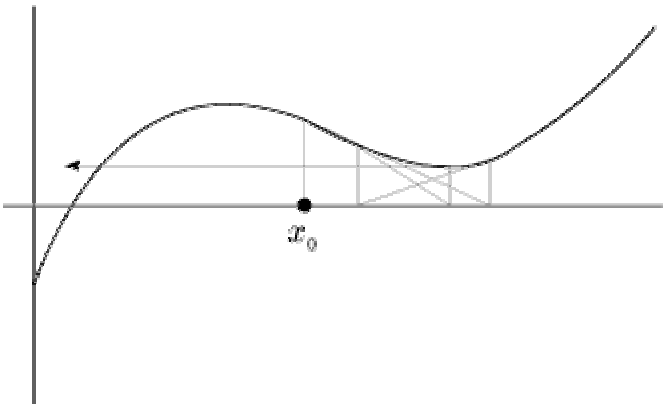
OBSERVACIONES.

1. Es conveniente dar de inicio, una buena aproximación inicial. Se aclara este hecho al destacar el error en el Polinomio de Taylor, durante el proceso de derivación de la fórmula se consideró: $\left(x - \bar{x}^2\right)^2 \to 0$.

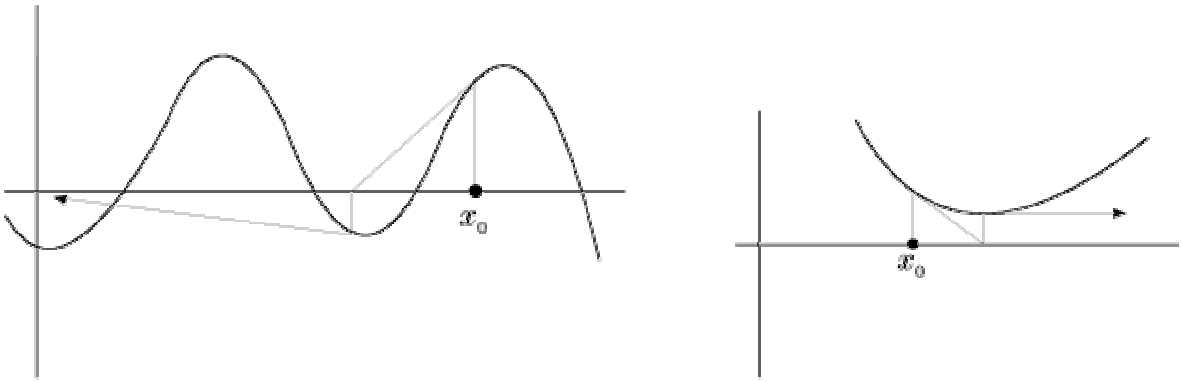
2. Cuando en las iteraciones hacia la raíz, ocurre que cualquiera de ellas esta cerca de un punto de inflexión, entonces la sucesivas iteraciones se alejan de la raíz.



3. El método requiere que $f'(\bar{x}) \neq 0$. En consecuencia, cuando en una iteración se presente esta situación, la pendiente de la recta tangente a la curva en dicho punto es paralela al eje X, en consecuencia el Método de Newton-Raphson diverge.



4. Puede ocurrir que aunque se de un valor inicial cercano a una raíz, el método converja a otra que este más distante. Gráficamente:



5. El Método de Newton-Raphson tiene una rápida convergencia, a ser comparado con los anteriores.

Teorema 2.2.5.1: Sea $f \in C^2[a,b]$. Si $p \in [a,b]$ es tal que *f(p) = 0* y $f'(p) \neq 0$, entonces existe $\delta > 0$ tal que el método de Newton genera una sucesión $\{P_n\}_{n=1}^{\infty}$, que converge a *p* para cualquier aproximación inicial $p_0 \in [p-\delta, p+\delta]$

Demostración:

Al considerar $P_n = P_{n-1} - \dfrac{f(P_{n-1})}{f'(P_{n-1})}$, se puede entonces considerar al proceso iterativo de Newton como un proceso de iteración funcional (punto fijo), basta considerar $g(P_{n-1}) = P_{n-1} - \dfrac{f(P_{n-1})}{f'(P_{n-1})}; (n \geq 1)$. Luego $P_n = g(P_{n-1})$. Así que el problema entonces se limita a

encontrar un $K \in (0,1)$, que el intervalo $[p-\delta, p+\delta]$ es mapeado en sí mismo $(g(x) \in [p-\delta, p+\delta])$ y que $|g'(x)| \le K < 1, \forall x \in [p-\delta, p+\delta]$. (De acuerdo al Teorema del Punto Fijo). Puesto que *f'(p)≠ 0* y *f'* es continua, $\exists \delta_1 > 0 \ni f'(x) \neq 0$, $\forall x \in [p-\delta_1, p+\delta_1] \subset [a,b]$. Por otra parte como $g(x) = x - \frac{f(x)}{f'(x)}$; por ser f y f' continuas en [*a,b*]; en particular g será continua en $[p-\delta_1, p+\delta_1]$. Adicionalmente,

$$g'(x) = 1 - \frac{f'(x)\cdot f'(x) - f(x)\cdot f''(x)}{[f'(x)]^2}$$

$$g'(x) = \frac{[f'(x)]^2 - [f'(x)]^2 + f(x)\cdot f''(x)}{[f'(x)]^2} = \frac{f(x)\cdot f''(x)}{[f'(x)]^2}$$; o sea:

$g'(x) = \frac{f(x)\cdot f''(x)}{[f'(x)]^2}$; para $x \in [p-\delta_1, p+\delta_1]$; ahora bien:

$f \in C^2[a,b]$, entonces $g \in C^1[p-\delta_1, p+\delta_1]$. Pero por ser *f(p)= 0* (hipótesis):

$g'(p) = \frac{f(p)\cdot f''(p)}{[f'(p)]^2} = 0$; y puesto que *g'* es continua (por serlo *f*, *f'* y *f''*), entonces para cualquier *K*<1, $\exists \delta$ tal que $0 < \delta < \delta_1 \wedge |g'(x)| \le K, \forall x \in [p-\delta, p+\delta]$.

Demostremos que *g* mapea $[p-\delta, p+\delta]$ en el mismo intervalo. Si $x \in [p-\delta, p+\delta]$, por el Teorema del Valor Medio, se tiene que para algún ξ entre $x \wedge p$, se tiene que $|g(x)-g(p)| = |g'(\xi)(x-p)|$ por lo tanto $|g(x)-p| = |g(x)-g(p) = |g'(\xi)||x-p| \le K|x-p| < |x-p| < \delta$ por lo que $g(x) \in [p-\delta, p+\delta]$ por lo tanto; por el Teorema de Punto Fijo:

Con $g(x) = x - \frac{f(x)}{f'(x)}$, la sucesión $\{P_n\}_{n=1}^{\infty}$ definida como $P_n = g(P_{n-1}), n \ge 1$ converge a *p*, para cualquier $P_o \in [p-\delta, p+\delta]$

Ejemplo 2.2.5.2 : Verifiquemos que en el problema resuelto por Leonardo, si se le aplica el Método de Newton, la raíz se consigue más rápido. Como $f(x) = x^3 + 2x^2 + 10x - 20$, en [1,2], cumple con las hipótesis para aplicar éste método y con $f'(x) = 3x^2 + 4x + 10$, se obtiene tomando el mismo punto de inicio (1), el siguiente resultado:

I	X1	F(X1)	F'(X1)	FP
1	1.0000000000	-7.0000000000	17.0000000000	1.4117647059
2	1.4117647059	0.9175656422	21.6262975780	1.3693364706
3	1.3693364706	0.0111481194	21.1025929910	1.3688081886

4 1.3688081886 0.0000017045 21.0961403260 1.3688081078
5 1.3688081078 0.0000000000 21.0961393390 1.3688081078

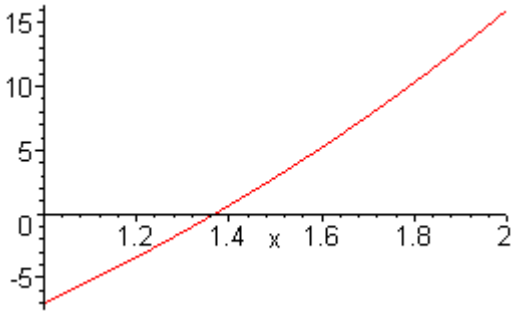
RAIZ APROXIMADA = 1.3688081078 LOGRADA CON 5 ITERACIONES

Apoyándonos en Maple V®, la gráfica de *f*, en el intervalo [1,2] y su derivada se consiguen con las instrucciones siguientes:

```
> f:=x^3+2*x^2+10*x-20;

> plot(f(x), x=1..2);

> diff(f,x);
```



2.3. BIBLIOGRAFÍA COMPLEMENTARIA

[1] BURDEN, R. – FAIRES, J. *ANALISIS NUMERICO,* 6ta Ed. Revisada. International Thompson Editores. 1999.

[2] CHAPRA, S. & CANALE, R. Métodos Numéricos. 3ra. Ed. McGraw-Hill. México. 2000.

[3] SCHEID, F. *ANALISIS NUMERICO*. McGraw-Hill. México. 1972.