MECÁNICA COMPUTACIONAL I

Capítulo 2

Solución de ecuaciones no –lineales

Contenido

- Búsqueda de raíces: introducción
- Métodos gráficos
- Métodos cerrados: Teorema de Bolzano. Criterios de parada. Método de la bisección. Interpolación lineal (normal y modificado). Criterios de convergencia y análisis de errores.
- Métodos abiertos: Newton–Raphson. Método de punto fijo. Criterios de parada. Aceleración de Aitken: método de Steffensen.
- Método de la secante. Newton relajado. Newton modificado.

Búsqueda de Raíces: Introducción

¿Cómo hallar la raíz de una ecuación no lineal o trascendente?

$$f(x) = 0$$

Toda función de una variable puede ser expresada como f(x)=0.

Por ejemplo hallar x que satisface

$$sen(x) = x$$

se puede expresar como

$$f(x) = sen(x) - x = 0$$

Búsqueda de Raíces: Generalidades

Cuando se desea hallar raíces de ecuaciones, ciertas preguntas deben formularse respecto al método a utilizar:

¿Se buscará la raíz una sola vez o deberá buscarse varias veces?

¿Qué grado de precisión se desea?

¿Qué tan rápido debe ser el método que empleemos?

¿Debe ser robusto?

¿La función es un polinomio?

¿La función es continua o posee singularidades?

Dependiendo de las repuestas a las preguntas anteriores se escogerá el método.

En general no existe un método que se adapte a todas las circunstancias.

Tres clases de métodos están disponibles

a)Métodos gráficos

Una gráfica permite estimar la ubicación de la raíz

b)Métodos cerrados

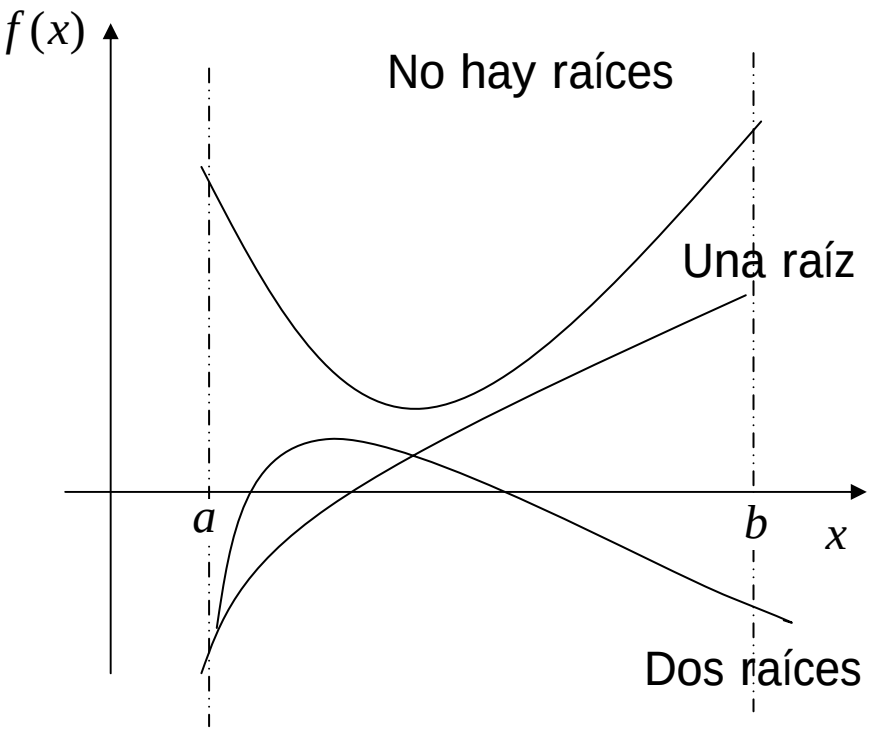
La raíz se busca dentro de un intervalo particular

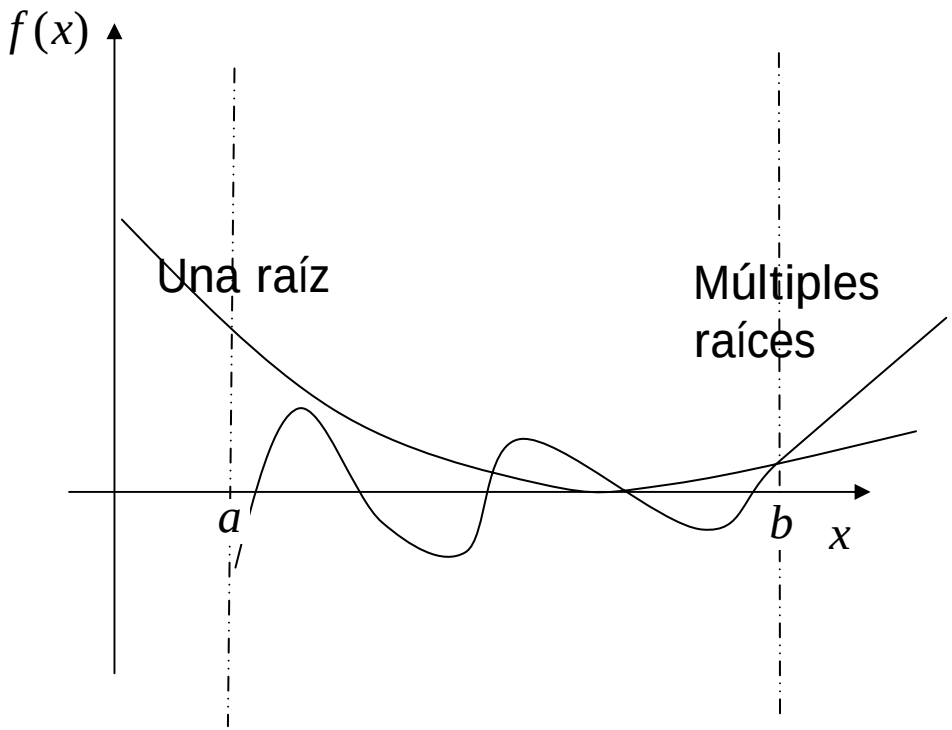
c)Métodos abiertos

La raíz se busca a partir de un punto particular

Utilizando cualquier procedimiento se traza una gráfica de la función. La raíz se selecciona visualmente.

Diversas situaciones se presentan

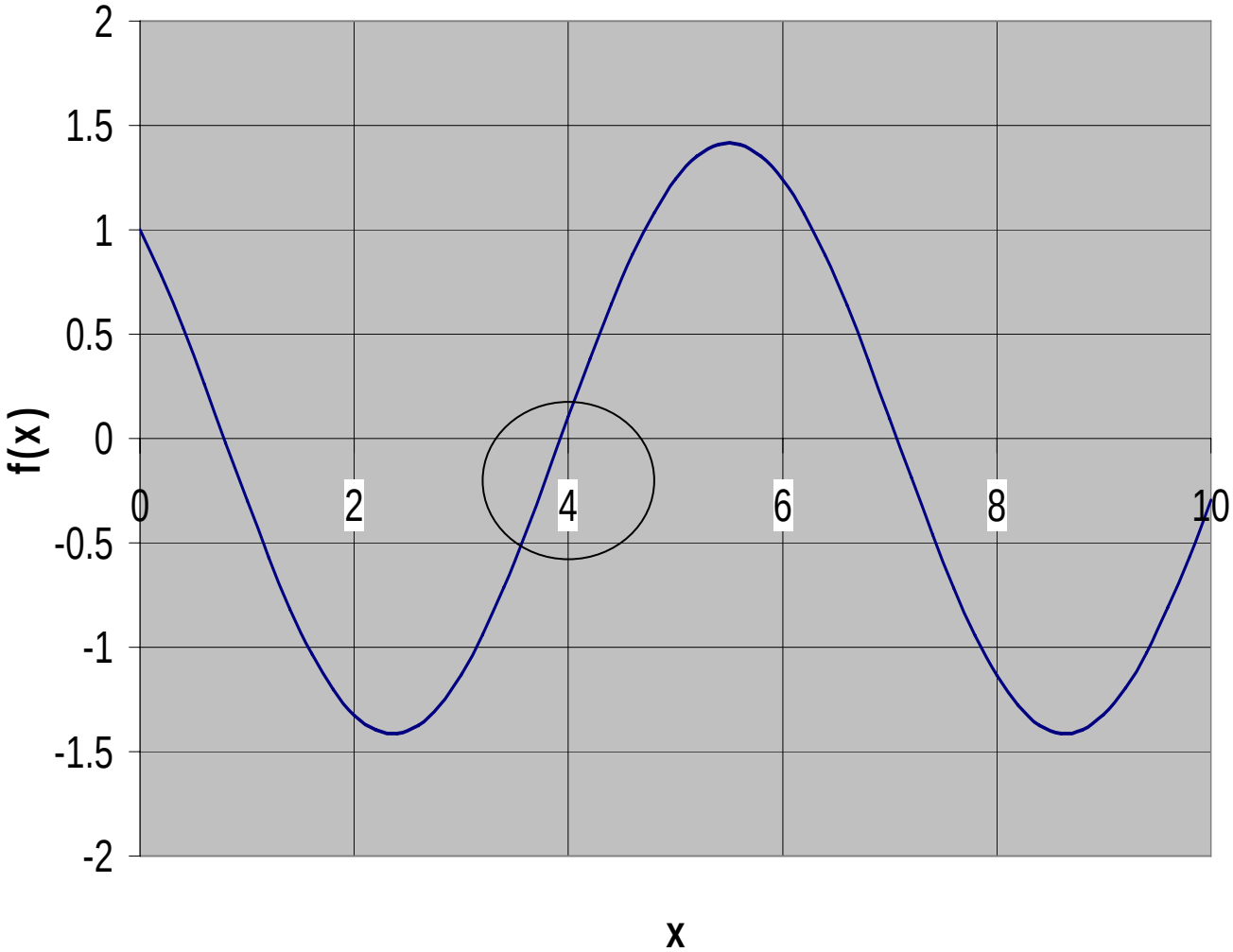




Las condiciones particulares de la situación bajo análisis determinará la raíz a buscar

Ejemplo: hallar la raíz de $f(x) = \cos(x) - sen(x) = 0$

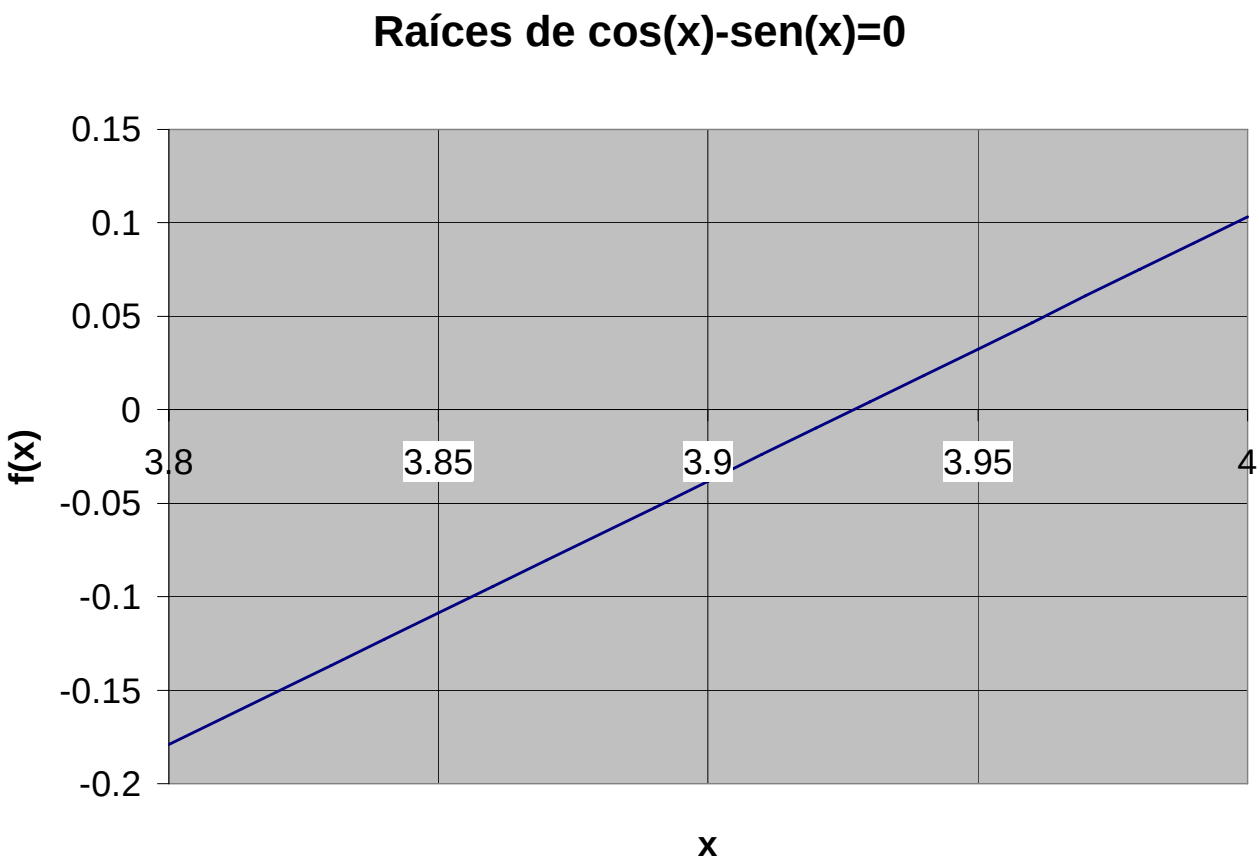
x	f(x)
0	1
0.2	0.78139725
0.4	0.53164265
0.6	0.26069314
0.8	-0.02064938
1	-0.30116868
1.2	-0.56968133
1.4	-0.81548259
1.6	-1.02877313
1.8	-1.20104973
2	-1.32544426
2.2	-1.39699752
2.4	-1.4128569
2.6	-1.37239013
2.8	-1.27721049
3	-1.1311125
3.2	-0.93992063
3.4	-0.71125709
3.6	-0.45423797
3.8	-0.17910982
4	0.10315887



Supongamos que la raíz de interés está cercana a 4

Luego, en el nuevo intervalo de búsqueda

x	f(x)
3.8	-0.17910982
3.81	-0.16507284
3.82	-0.15101936
3.83	-0.13695077
3.84	-0.12286849
3.85	-0.10877392
3.86	-0.09466848
3.87	-0.08055356
3.88	-0.0664306
3.89	-0.05230099
3.9	-0.03816615
3.91	-0.02402749
3.92	-0.00988643
3.93	0.00425562
3.94	0.01839724
3.95	0.03253703
3.96	0.04667356
3.97	0.06080542
3.98	0.0749312
3.99	0.08904949
4	0.10315887



El proceso continua hasta que se “encuentra” la raíz.

El método gráfico permite obtener fácilmente una primera aproximación a la raíz, sin embargo es muy poco preciso.

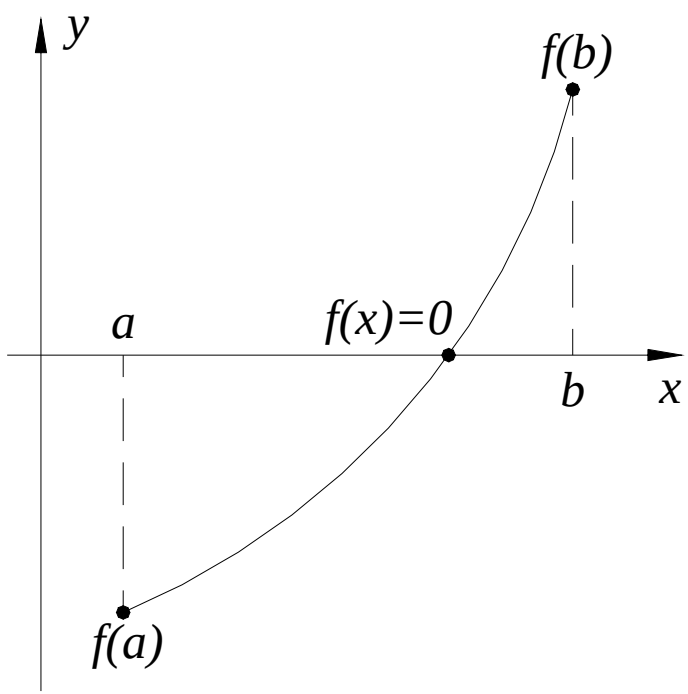
No obstante, permite hallar rápidamente un intervalo en el cual realizar la búsqueda.

Estos intervalos pueden ser utilizados por métodos cerrados, los cuales son capaces de encontrar la raíz, de manera mas eficiente.

Los métodos cerrados se basan en el teorema de Bolzano el cual se desarrolla a continuación.

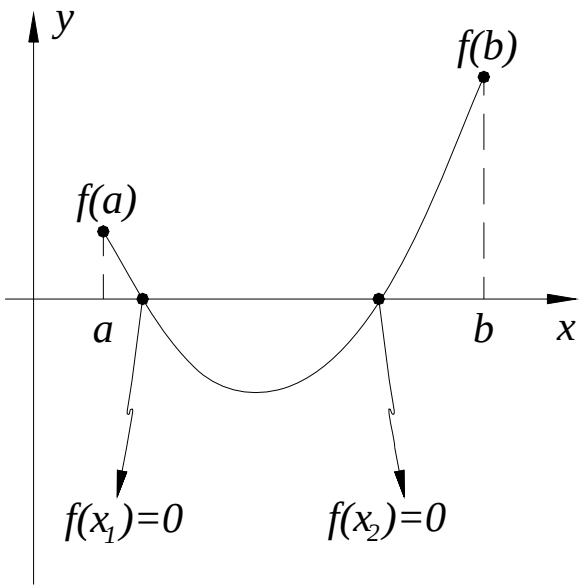
Teorema de Bolzano

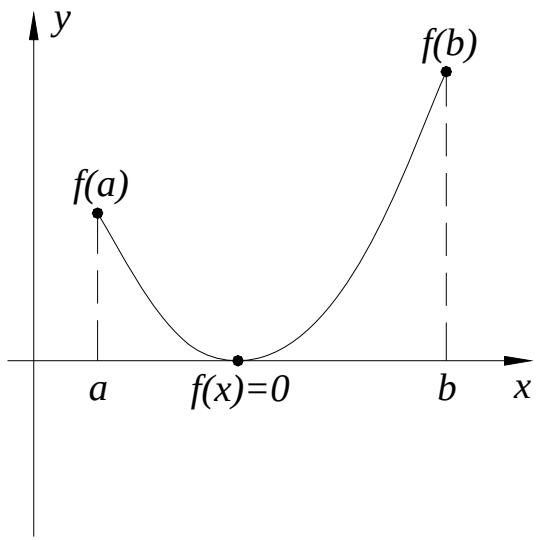
Teorema de Bolzano: dado un intervalo cerrado [a,b] y una función continua f(x), existe al menos una solución a la ecuación trascendente f(x)=0 si $f(a)*f(b) \leq 0$.



Teorema de Bolzano

El teorema de Bolzano garantiza la existencia de una raíz si existe un cambio de signo en el intervalo [a,b], pero la antítesis es falsa, sino existe un cambio de signo, también puede existir una raíz en [a,b].





Los métodos numéricos que están basados en el teorema de Bolzano se denominan "*métodos cerrados*", ya que exigen como argumento un intervalo cerrado donde la función experimente un cambio de signo.

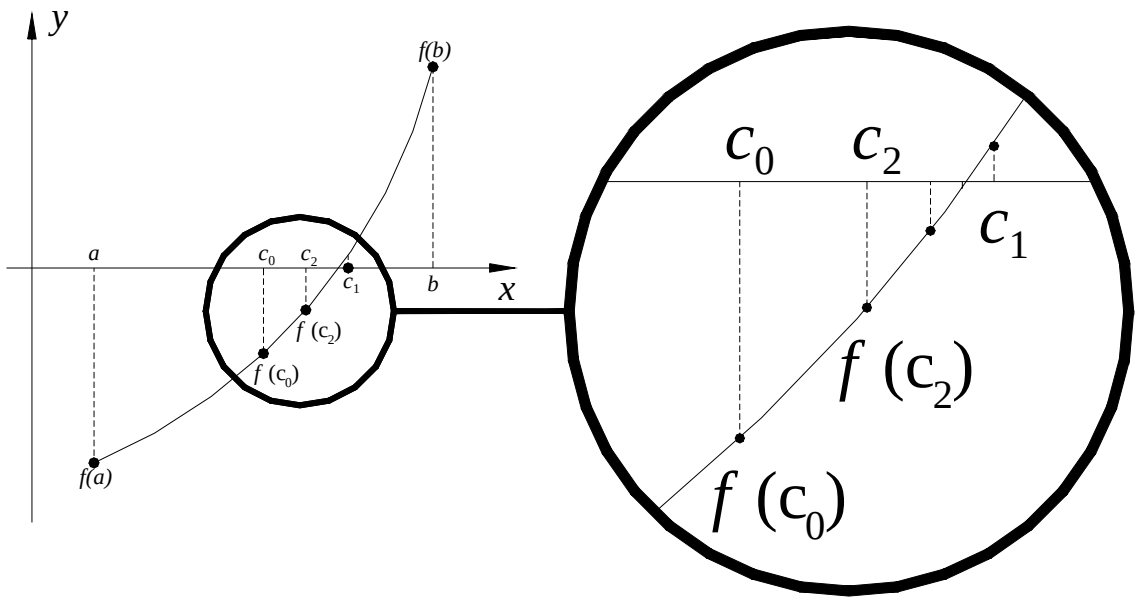
Suponga que una función continua *f* definida sobre el intervalo [a,b] es conocida, con f(a) y f(b) de signo opuesto.

Entonces según el teorema de Bolzano existe x en [a,b] tal que f(x)=0. Este método divide el intervalo cerrado inicial en dos mitades en cada paso, ubicando después a la mitad que contiene a *x*.

Esto significa que un estimado mejorado que se aproxima a la raíz viene dado por c tal que

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

Método de la Bisección



El algoritmo implementará este método introduciendo instrucciones como:

```
SI (f(a) * f(c) < 0) b = c;
SINO                 a = c;
```

- Luego el algoritmo continúa ejecutándose, dividiendo cada intervalo cerrado que contenga a la raíz en dos mitades y eligiendo aquella que satisface el teorema de Bolzano.
- ¿Cuándo debe detenerse este proceso iterativo?.
- Para responder esta interrogante y proponer un algoritmo completo para el método de la bisección, debe hacerse un breve paréntesis en esta discusión para tratar en detalle a los criterios de parada de los algoritmos para resolver ecuaciones no lineales.

Criterios de parada

Los distintos criterios de parada se enumeran a continuación:

–**Error Local**: $\varepsilon_{n+1}=|x_{n+1} - x_n|$, donde x_n es la n-ésima aproximación a raíz *x*.

–**Desviación**: $\delta_n = |f(x_n)|$.

–**Número máximo de iteraciones** (N^{Max}) o criterio de seguridad.

El *error local* compara los valores de iteraciones sucesivas, es decir mide la precisión del esquema iterativo.

Si el algoritmo esta convergiendo, esto es aproximándose a la solución, el error local debe disminuir enfatizando que existe una marcada tendencia a la repetición producto de la convergencia.

Sin embargo debe verificarse por otra vía que no existe un error sistemático, para ello debe emplearse a la *desviación*, esta mide directamente que tan alejada puede estar una aproximación a la raíz.

Finalmente, el *criterio de seguridad* o del número máximo de iteraciones acota el proceso iterativo restringiéndolo estrictamente a una cantidad definida de iteraciones, este criterio permite detener al algoritmo cuando se alcanza un limite de iteraciones permisibles.

Los dos primeros criterios tienen una tolerancia asociada, esta referencia al valor aceptable o razonable por debajo del cual puede considerarse que se ha alcanzado una solución que satisface las expectativas de un usuario del algoritmo.

Criterios de parada

Estas tolerancias son constantes reales positivas denotadas en lo sucesivo, *tolerancia de error local* ε^{TOL}, y *tolerancia de desviación*, δ^{TOL} .

El *número máximo de iteraciones*, N^{Max}, es un numero natural mayor de cero.

Algoritmo del método de la Bisección

```
Entrada de datos: a, b, f(x), ε^Tol, δ^Tol, N^Máx
Salida de datos: aproximación de la raíz x o un mensaje del error.
//Verificar si se satisface el teorema de Bolzano
SI (f(a) * f(b) < 0)
    //Inicialización de variables
    n = 0;
    //Entrada en el bucle de cálculo
    HACER
        //Calcular la aproximación a la raíz
        c = (a + b) / 2.0;
        //Verificar en que mitad esta la raíz
        SI (f(a) * f(c) < 0) b = c; //x ∈ I[a, c]
        SINO                 a = c; //x ∈ I[c, b]
        //Incrementar el contador de control
        n = n + 1;
    MIENTRAS (((0.5 * |b - a| > ε^Tol) O (|f(c)| > δ^Tol)) Y (n < N^Máx);
    //Verificar si se encontró la solución
    SI (n < N^Máx)
        Imprimir c;
    SINO
        Imprimir "No se satisfacen las tolerancias...";
    FIN SI
SINO //Negación del Teorema de Bolzano
    Imprimir "No se satisface el teorema de Bolzano.";
FIN SI
```

Ejemplo del método de la Bisección

Resuélvase la siguiente ecuación trascendente empleando el método de la bisección, en el intervalo [0,1]. El error máximo permitido es de 0.0005

$$f\left(x\right)=6x^3-5x^2+7x-2=0$$

Los resultados de la ejecución del algoritmo del método de la bisección se muestran en la tabla.

n	x1	x2	f(x1)	f(x2)	x3	f(x3)	Prueba	Error
1	0	1	-2	6	0.5	1	0	0.5
2	0	0.5	-2	1	0.25	-0.46875	1	0.25
3	0.25	0.5	-0.46875	1	0.375	0.23828125	0	0.125
4	0.25	0.375	-0.46875	0.23828125	0.3125	-0.117675781	1	0.0625
5	0.3125	0.375	-0.117675781	0.23828125	0.34375	0.059143066	0	0.03125
6	0.3125	0.34375	-0.117675781	0.059143066	0.328125	-0.02948761	1	0.015625
7	0.328125	0.34375	-0.02948761	0.059143066	0.3359375	0.014763832	0	0.0078125
8	0.328125	0.3359375	-0.02948761	0.014763832	0.33203125	-0.00737679	1	0.00390625
9	0.33203125	0.3359375	-0.00737679	0.014763832	0.333984375	0.003689662	0	0.001953125
10	0.33203125	0.333984375	-0.00737679	0.003689662	0.333007813	-0.001844512	1	0.000976563

Error máximo en el método de la Bisección

Una aproximación del error máximo cometido al utilizar el método de bisección es obtenido muy fácilmente al considerar que, inicialmente, el error máximo es dado por

$$\varepsilon^{(1)} = \frac{(b-a)}{2} = \frac{(b-a)}{2^1}$$

El error máximo luego de la primera iteración el error máximo es:

$$\varepsilon^{(2)} = \frac{\frac{(b-a)}{2}}{2} = \frac{(b-a)}{2^2}$$

Luego de n iteraciones el error máximo es dado por:

$$\varepsilon^{(n)} = \frac{\frac{(b-a)}{2}}{2} = \frac{(b-a)}{2^n}$$

Error máximo en el método de la Bisección

En consecuencia, aún cuando no se conoce la raíz exacta de la ecuación f(x)=0, es posible calcular fácilmente una cota para el error.

Además, podemos fácilmente estimar la cantidad de iteraciones necesarias para obtener la raíz con una cota de error dada ε a partir de:

$$\frac{(b-a)}{2^n} \leq \varepsilon$$

$$\frac{(b-a)}{\varepsilon} \leq 2^n$$

$$n \geq \frac{\ln\left(\frac{(b-a)}{\varepsilon}\right)}{\ln 2}$$

Error máximo en el método de la Bisección

En nuestro ejemplo anterior, si se quisiera calcular, el número de iteraciones necesarias para hallar la raíz con un error máximo de 0.01 tendríamos

$$n \geq \frac{\ln\left(\frac{(b-a)}{\varepsilon}\right)}{\ln 2} = \frac{\ln\left(\frac{(1-0)}{0.01}\right)}{\ln 2} = 6.64$$

Luego, puesto que n es entero tendremos $n = 7$

n	x1	x2	f(x1)	f(x2)	x3	f(x3)	Prueba	Error
1	0	1	-2	6	0.5	1	0	0.5
2	0	0.5	-2	1	0.25	-0.46875	1	0.25
3	0.25	0.5	-0.46875	1	0.375	0.23828125	0	0.125
4	0.25	0.375	-0.46875	0.23828125	0.3125	-0.117675781	1	0.0625
5	0.3125	0.375	-0.117675781	0.23828125	0.34375	0.059143066	0	0.03125
6	0.3125	0.34375	-0.117675781	0.059143066	0.328125	-0.02948761	1	0.015625
8	0.328125	0.3359375	-0.02948761	0.014763832	0.33203125	-0.00737679	1	0.00390625
9	0.33203125	0.3359375	-0.00737679	0.014763832	0.333984375	0.003689662	0	0.001953125
10	0.33203125	0.333984375	-0.00737679	0.003689662	0.333007813	-0.001844512	1	0.000976563

Método de Interpolación lineal

El método de la bisección es relativamente sencillo de implementar y el análisis de error es bastante obvio, sin embargo, no es muy eficiente.

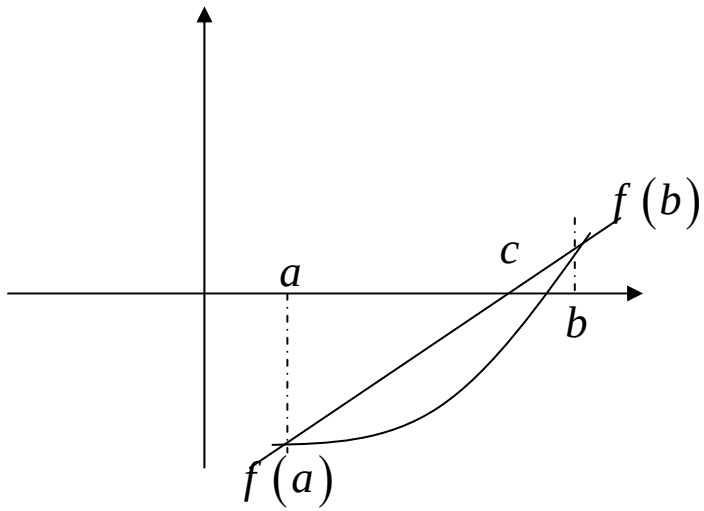
Para la mayoría de las funciones podemos mejorar la tasa de convergencia a la raíz.

El método de la bisección se fundamenta en la división del intervalo [a,b] en mitades iguales, sin embargo, no se considera las magnitudes de las ordenadas f(a) y f(b).

Por ejemplo, si f(a) es más cercana a cero que f(b), es más probable que la raíz esté más cercana de la abscisa *a* que de *b*.

Método de Interpolación lineal

El método de la interpolación lineal consiste precisamente en unir dichos puntos con una línea recta, como se muestra en la figura.

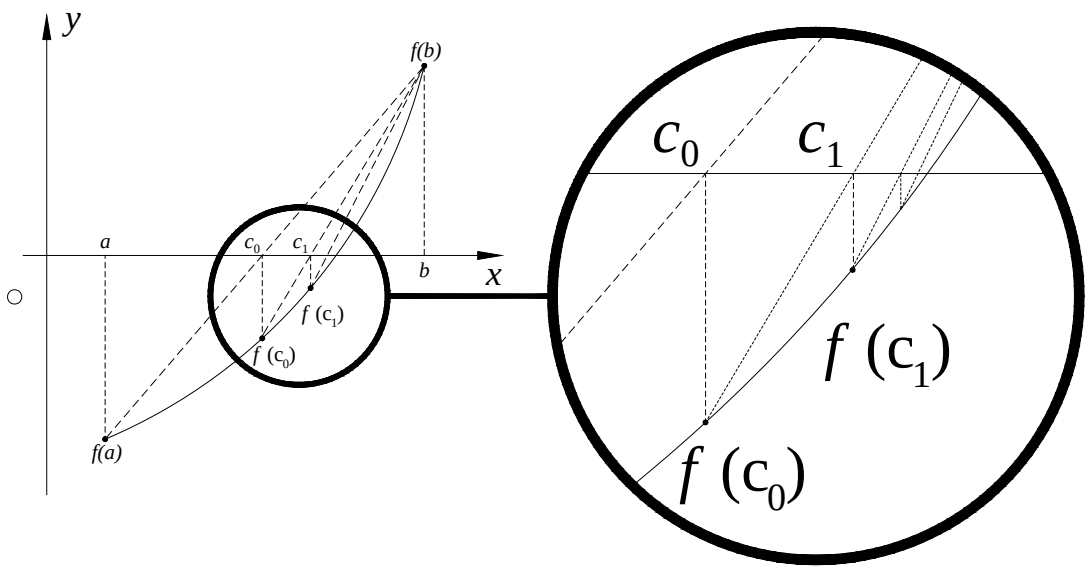


La intersección de ésta recta con el eje de las abscisas representa el estimado mejorado de la raíz, es decir:

$$y - f(b) = \frac{f(b) - f(a)}{(b-a)}(x-b)\Big|_{\substack{y=0 \\ x=c}} \Rightarrow c = b - \frac{f(b)}{f(b) - f(a)}(b-a)$$

Luego de obtener este estimado, se evalúa la función en ese punto, f(c), verificando después cual intervalo satisface el teorema de Bolzano, análogamente al procedimiento descrito en el método de bisección:

```
SI (f(a) * f(c) < 0) b = c;
SINO                 a = c;
```



Algoritmo del Método de Interpolación lineal

```
Entrada de datos: a, b, f(x), ε^Tol, δ^Tol, N^Máx
Salida de datos: aproximación de la raíz x o un mensaje del error.
//Almacenar los valores de ordenada en dos variables auxiliares
Fa = f(a), Fb = f(b);
//Verificar si se satisface el teorema de Bolzano
SI (Fa * Fb < 0)
    //Inicialización de variables
    n = 0, Cold = (a + b) / 2.0;
    //Entrada en el bucle de cálculo
    HACER
        //Calcular la aproximación a la raíz
        c = b - (Fb * (b - a)) / (Fb - Fa), Fc = f(c);
        //Verificar en que mitad esta la raíz
        SI (Fa * Fc < 0) b = c, Fb = Fc; //x ∈ I[a, c]
        SINO             a = c, Fa = Fc; //x ∈ I[c, b]
        //Calcular el error local
        ErrorL = |c - Cold|;
        //Incrementar el contador de control y actualizar a Cold
        n = n + 1, Cold = c;
    MIENTRAS (((ErrorL > ε^Tol) O (|Fc| > δ^Tol)) Y (n < N^Máx);
    //Verificar si se encontró la solución
    SI (n < N^Máx)
        Imprimir c;
    SINO
        Imprimir "No se satisfacen las tolerancias...";
    FIN SI
SINO //Negación del Teorema de Bolzano
    Imprimir "No se satisface el teorema de Bolzano.";
```

Ejemplo del método de Interpolación lineal

Resuélvase la siguiente ecuación empleando el método de interpolación lineal. Busque la raíz en el intervalo [0;1.3]

$$f\left(x\right)=x^{10}-1=0$$

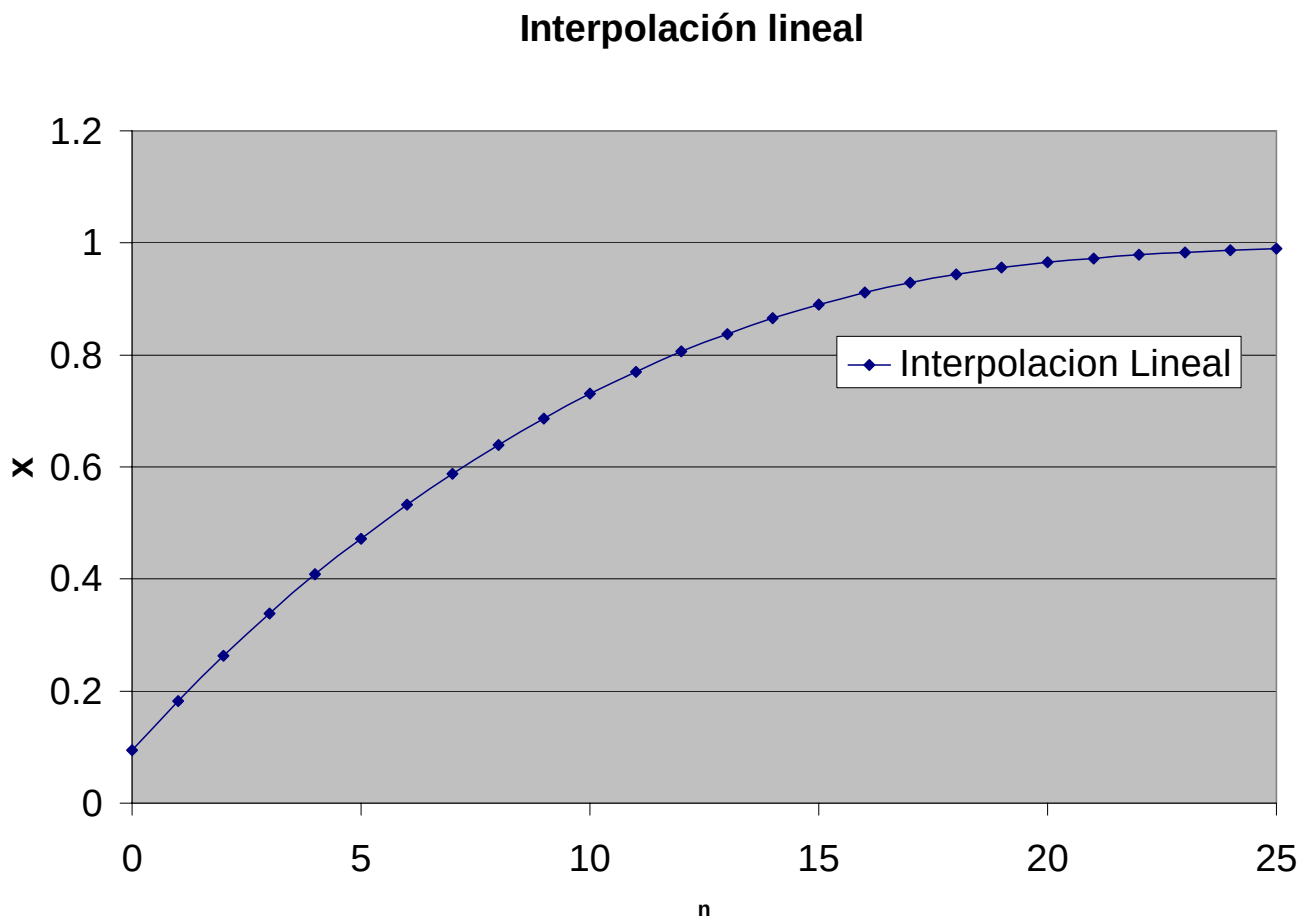
Los resultados de la ejecución del algoritmo del método de interpolación lineal se muestran en la tabla

n	x1	x2	f(x1)	f(x2)	x3	f(x3)
0	0	1.3	-1	12.7858492	0.0942996	-1
1	0.0942996	1.3	-1	12.7858492	0.18175887	-0.99999996
2	0.18175887	1.3	-0.99999996	12.7858492	0.26287401	-0.99999842
3	0.26287401	1.3	-0.99999842	12.7858492	0.3381051	-0.99998048
4	0.3381051	1.3	-0.99998048	12.7858492	0.40787792	-0.99987256
5	0.40787792	1.3	-0.99987256	12.7858492	0.47258315	-0.99944437
6	0.47258315	1.3	-0.99944437	12.7858492	0.53257151	-0.99816439
7	0.53257151	1.3	-0.99816439	12.7858492	0.58814457	-0.99504731
8	0.58814457	1.3	-0.99504731	12.7858492	0.63954397	-0.98855267
9	0.63954397	1.3	-0.98855267	12.7858492	0.68694317	-0.97660042
10	0.68694317	1.3	-0.97660042	12.7858492	0.73044644	-0.95676019

Ejemplo del método de Interpolación lineal

Gráficamente

$$f\left(x\right)=x^{10}-1=0$$

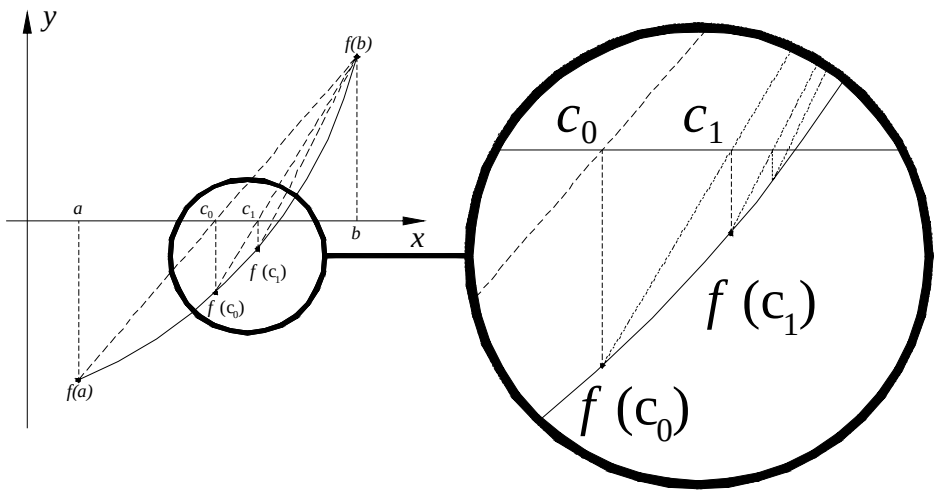


Método de Interpolación lineal modificada

Convergencia unilateral

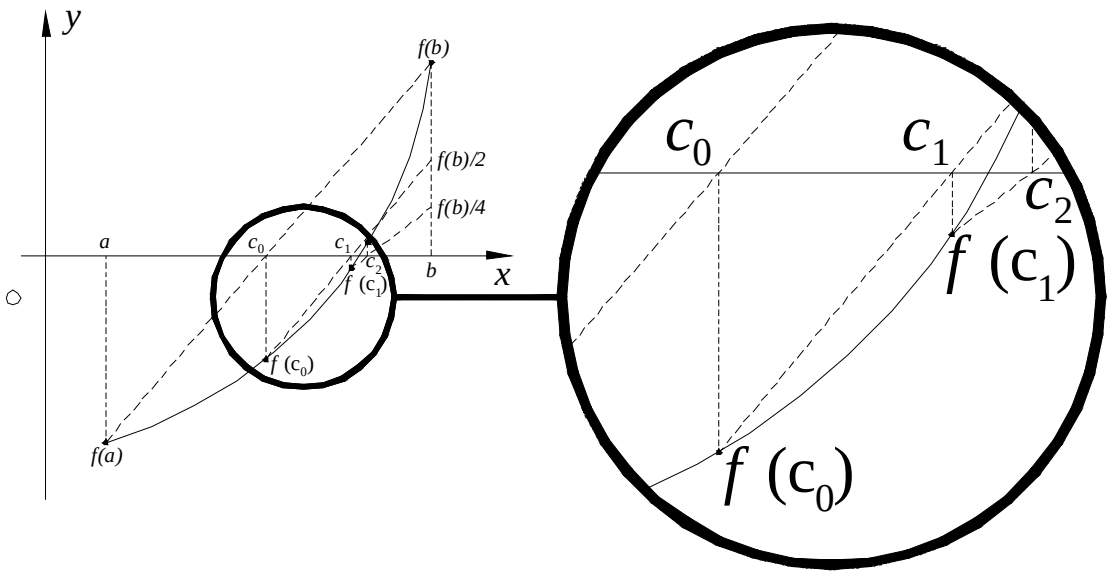
El ejemplo resuelto en clase destaca una seria falla del método de la interpolación lineal, el criterio de aproximación a la raíz es unilateral.

Si la función tiene una curvatura significativa entre *a* y *b*, esto puede perjudicar a la tasa de convergencia a la raíz.



Método de la interpolación lineal modificada:

Un remedio para la convergencia unilateral consiste en reemplazar el valor de f(x) y de la posición final estancada por f(x)/2 .



Método de Interpolación lineal modificada

Entrada de datos: a, b, $f(x)=0$, E^{Tol}, δ^{Tol}, $n^{máx}$.

Salida de datos: Aproximaciones a la raíz x o mensaje de error.

//Verificar si se satisface el teorema de Bolzano.

si $(f(a)\cdot f(b)<0)$

//Inicialización:

$n=0$, $c_{OLD}=\left(\frac{a+b}{2}\right)$, $Fa=f(a)$, $Fb=f(b)$, $SAVE=f(a)$;

Hacer

$c=b-f(b)\frac{Fb\cdot(b-a)}{Fb-Fa}$, $Fc=f(c)$;

// Calcular el error local

$ErrorL=|c-c_{OLD}|$;

// Verificar Bolzano

Si $(Fa\cdot Fc<0)$ // $x\in I=[a,c]$

$b=c$, $Fb=Fc$;

Si $(Fc\cdot SAVE>0)$ $Fa=Fa/2$;

Sino // $x\in I=[c,b]$

$a=c$, $Fa=Fc$;

Si $(Fc\cdot SAVE>0)$ $Fb=Fb/2$;

Fin Si

Ejemplo: Método de Interpolación lineal modificada

Continuación...

// Incrementar el contador y almacenar estimando

$n = n + 1 \quad , \quad c_{OLD} = c \quad , \quad SAVE = Fc \;;$

Mientras $\left(\left(\left(ErrorL > E^{Tol}\right) o \left(|f(c)| > \delta^{Tol}\right)\right) y \left(n < n^{máx}\right)\right)$

//Verificar si se obtuvo la solución

Si $\left(n < n^{máx}\right)$ Imprimir 'c';

Sino Imprimir "No se Satisface las tolerancias";

Sino

Imprimir "no se satisface el teorema de Bolzano"; Imprimir "Especifique otro intervalo";

Fin Si

Método de Interpolación lineal modificada

Resuélvase la siguiente ecuación trascendente empleando el método de interpolación lineal modificado. Busque la raíz en el intervalo [0;1.3]

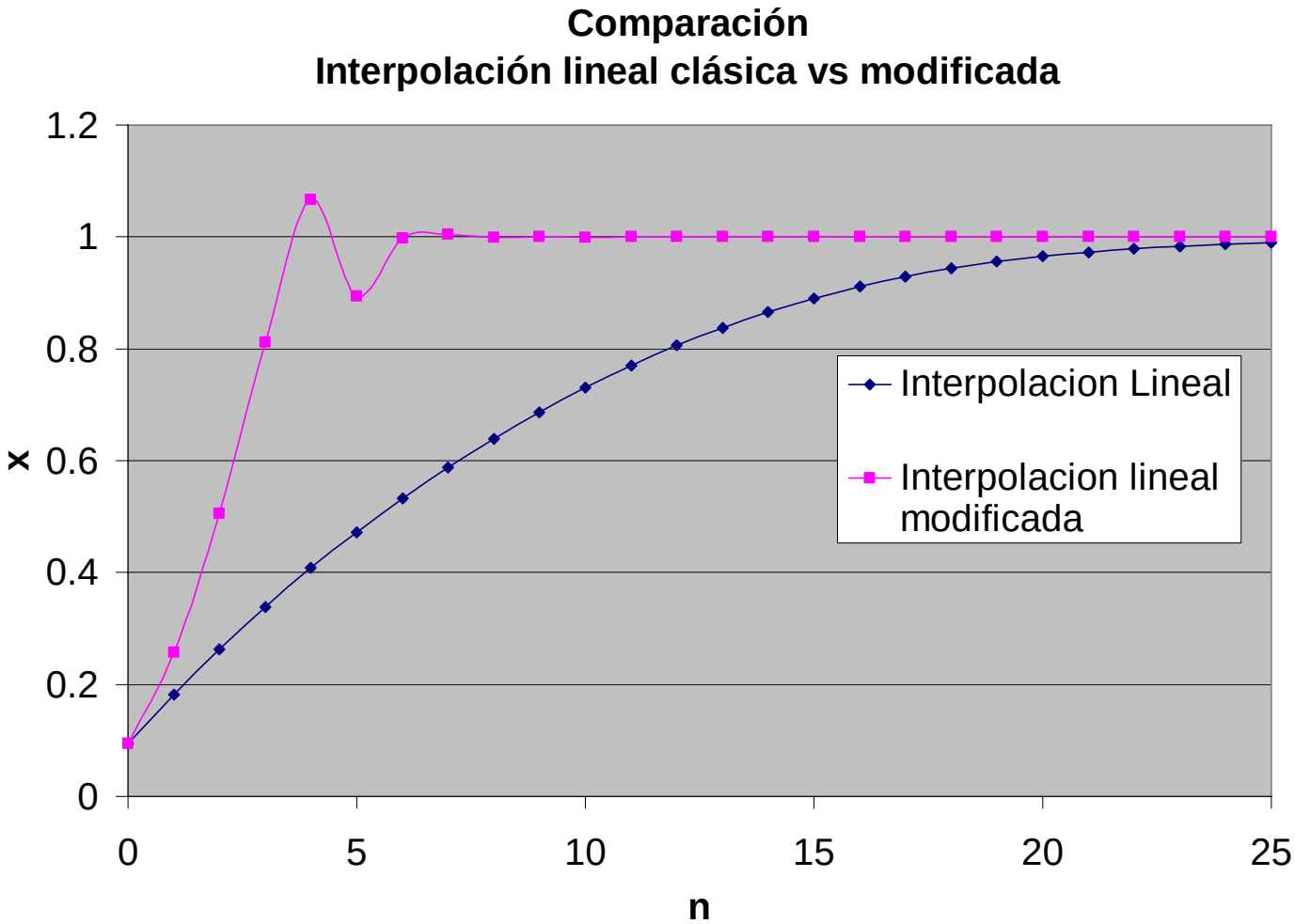
$$f\left(x\right)=x^{10}-1=0$$

Los resultados de la ejecución del algoritmo del método de interpolación lineal modificada se muestran en la tabla

n	x1	x2	f(x1)	f(x2)	x3	f(x3)
0	0	1.3	-1	12.7858492	0.0942996	-1
1	0.0942996	1.3	-1	6.39292459	0.25738802	-0.99999872
2	0.25738802	1.3	-0.99999872	3.1964623	0.505838	-0.99890323
3	0.505838	1.3	-0.99890323	1.59823115	0.81128657	-0.87647841
4	0.81128657	1.3	-0.87647841	0.79911557	1.06692533	0.91135019
5	0.81128657	1.06692533	-0.43823921	0.91135019	0.89429769	-0.67279413
6	0.89429769	1.06692533	-0.67279413	0.4556751	0.99721841	-0.02747031
7	0.99721841	1.06692533	-0.02747031	0.22783755	1.00471865	0.04820117
8	0.99721841	1.00471865	-0.01373515	0.04820117	0.99888168	-0.01112708
9	0.99888168	1.00471865	-0.01112708	0.02410058	1.00072536	0.0072773
10	0.99888168	1.00072536	-0.00556354	0.0072773	0.99968049	-0.00319052

Método de Interpolación lineal modificada

La siguiente gráfica presenta la comparación entre ambos métodos



MATLAB: funciones (1)

itor - C:\MATLAB7\work\SolEcSegGr.m

Edit Text Cell Tools Debug Desktop Window Help

Stack: Base

1.0 1.1

```
function [x1 x2] = SolEcSegGr(a,b,c)
%  SolEcSegGr halla las raices de una ecuación de segundo grado
%  a x^2 + b x + c = 0
% a: coeficiente del término de segundo grado
% b: coeficiente del término de primer grado
% c: coeficiente del término independiente

% Calculo del discriminante

d = b^2 - 4*a*c;
if d>=0
    x1=(-b+sqrt(d))/(2*a);
    x2=(-b-sqrt(d))/(2*a);
    disp([' x1 =  ' num2str(x1)]);
    disp([' x2 =  ' num2str(x2)]);

else
    xr=-b/(2*a);
    xi=sqrt(abs(d))/(2*a);
    disp([' x1 =  ' num2str(xr)  ' + i ' num2str(xi)]);
    disp([' x2 =  ' num2str(xr)  ' - i ' num2str(xi)]);
    x1 = xr + i*xi;
    x2 = xr - i*xi;
end
```

Nombres coincidentes

Variables locales: sólo están definidas internamente en la función. Una vez llamada la función su valor no permanece a disposición

MATLAB: funciones (2)

```
function [x1 x2] = SolEcSegGr(a,b,c)
%  SolEcSegGr halla las raices de una ecuación de segundo grado
%  a x^2 + b x + c = 0
% a: coeficiente del término de segundo grado
% b: coeficiente del término de primer grado
% c: coeficiente del término independiente

% Calculo del discriminante

global d

d = b^2 - 4*a*c;

if d>=0
    x1=(-b+sqrt(d))/(2*a);
    x2=(-b-sqrt(d))/(2*a);
    disp([' x1 =  ' num2str(x1)]);
    disp([' x2 =  ' num2str(x2)]);

else
    xr=-b/(2*a);
    xi=sqrt(abs(d))/(2*a);
    disp([' x1 =  ' num2str(xr)  ' + i ' num2str(xi)]);
    disp([' x2 =  ' num2str(xr)  ' - i ' num2str(xi)]);
    x1 = xr + i*xi;
    x2 = xr - i*xi;
end
```

Variables globales: sólo están definidas en todas las funciones que contengan la expresión:
"global nombre_variable"

MATLAB: funciones (3)

```
function [c,n] = biseccion_2(a,b,e_tol,delta_tol,n_max_iter)
% Biseccion halla una raíz de una ecuación no lineal f(x)=0
% en el intervalo [a,b] basado en el método de bisección

% a :               Extremo inferior del intervalo de búsqueda
% b :               Extremo superior del intervalo de búsqueda
% e_tol :           Tolerancia permitida para el error local
% delta_tol :       Desviación permitida para la función
% n_max_iter :      Número máximo de iteraciones

func = inline('6*x^3 - 5*x^2 + 7*x - 2');
f_a = func(a);
f_b = func(b);

% Verificar si se satisface el teorema de Bolzano

if f_a*f_b < 0

    % Inicialización de variables
     n = 0;
    % calcular la primera aproximación a la raíz
    c=(a+b)/2;
    % f_c = 6*c^3 - 5*c^2 + 7*c -2;
    f_c = func(c);
    % Entrada en el bucle de cálculo
    while (((0.5*abs(b-a)>e_tol)|(abs(f_c) >delta_tol))&(n < n
       % calcular la aproximación a la raíz y evaluación de f(
       c=(a+b)/2;
       f_c = func(c);
        % verificar en que mitad esta la raíz
        if f_a*f_c < 0
            b=c;        % x está en [a,c]
            f_b = f_c;  % Asignación del valor de f(c) a f(b)
```

Funciones muy simples: inline.

Otras funciones mas complejas:
g = inline('sin(alpha*x)','x','alpha')

MATLAB: Escogiendo entre múltiples opciones

```
% Programa test_case
% Presenta estructuras para seleccionar opciones

color=input(' color = ');
if color==1
    disp('if Amarillo')
else
    if color==2
        disp('if Azul')
    else
        if color==3
            disp('if Rojo')
        else
            disp('if Negro')
        end
    end
end

if color==1
    disp('if_2 Amarillo')
elseif color==2
    disp('if_2 Azul')
elseif color==3
    disp('if_2 Rojo')
else color==4
    disp('if_2 Negro')
end
```

Si “anidado”

Si - sino

Casos

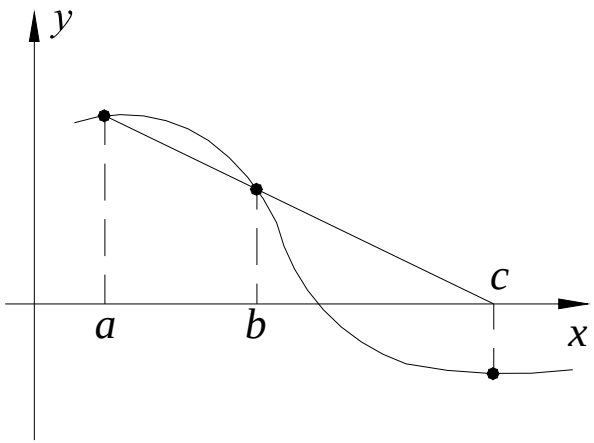
```
switch color
    case 1
        disp('case Amarillo')
    case 2
        disp('case Azul')
    case 3
        disp('case Rojo')
    otherwise
        disp('case Negro')
end
```

No cumplen necesariamente con el teorema de Bolzano.

Ventajas: tasa de convergencia mayor (más veloces).

Desventajas: no hay garantía de convergencia (muy importante el criterio de parada por seguridad).

Método de la Secante



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

Luego se debe actualizar la información

$a = b \quad , \quad b = c \quad ;$

Método de Newton-Raphson:

Si se desarrolla una función f(x) en serie de Taylor hasta orden 1 en torno a un valor dado $\bar{x}$ tenemos

$$f(x) = f(\bar{x}) + f'(\bar{x})(x - \bar{x}) + O(x - \bar{x})^2$$

Si x esta cerca de la raíz de f(x) entonces

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

Entonces

$$x \cong \bar{x} - \frac{f(\bar{x})}{f'(\bar{x})}$$

y un proceso iterativo puede ser llevado a cabo como

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

El algoritmo de Newton se usa ampliamente debido a que en la proximidad de la raíz, converge más rápidamente que los otros métodos vistos.

La derivada puede calcularse numéricamente a partir de

$$f'(x) \cong \frac{f(x+h) - f(x)}{h}; h << 1$$

en cuyo caso

$$x_{n+1} = x_n - \frac{h \cdot f(x_n)}{f(x_n + h) - f(x_n)}$$

Algoritmo método de Newton-Raphson

Entrada de datos: x_0, $f(x)$, $f'(x)$, ε^{Tol}, δ^{Tol}, $n^{máx}$

Salida de datos: aproximación a x o mensaje de error.

// Inicialización

$n = 0, \quad c_{OLD} = x_0$;

// Bucle de cálculo

Hacer

$$x_0 = x_0 - f(x_0) \Big/ f'(x_0) \ ;$$

$$ErrorL = |x_0 - c_{OLD}| \ ;$$

$$n = n + 1, \quad c_{OLD} = x_0 \ ;$$

Mientras $\left(\left(\left(ErrorL > \varepsilon^{Tol}\right) o \left(|f(x_o)| > \delta^{Tol}\right)\right) y \left(n < n^{máx}\right)\right)$

// Verificar si se obtuvo la solución

Si $\left(n < n^{máx}\right)$ imprimir x_0

Sino

Imprimir "No se satisface las tolerancias";

Imprimir "Especifique otra semilla de iteración";

Fin Si

Método de Newton-Raphson

Ejemplo. Determine la raíz de la ecuación trascendente

$$f(x) = x^3 + x^2 - 3x - 3 = 0$$

Determine la raíz en los alrededores de 0.5

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

x1 = 0.5

n	x0	f(x0)	f´(x0)	x1	f(x3)	Error (%)
0	0.5	-4.12500000	-1.25000000	-2.8	-8.71200000	117.85714286
1	-2.8	-8.71200000	14.92000000	-2.21609	-2.32398401	26.34889910
2	-2.21609	-2.32398401	7.30093712	-1.89777	-0.54004720	16.77298417
3	-1.89777	-0.54004720	4.00907807	-1.76307	-0.08271934	7.64044214
4	-1.76307	-0.08271934	2.79907828	-1.73351	-0.00372013	1.70476525
5	-1.73351	-0.00372013	2.54818645	-1.73205	-0.00000895	0.08428784
6	-1.73205	-0.00000895	2.53592800	-1.73205	0.00000000	0.00020376
7	-1.73205	0.00000000	2.53589839	-1.73205	0.00000000	0.00000000
8	-1.73205	0.00000000	2.53589838	-1.73205	0.00000000	0.00000000
9	-1.73205	0.00000000	2.53589838	-1.73205	0.00000000	0.00000000

Análisis del error en el método de Newton-Raphson

El método de Newton-Raphson parte del desarrollo en serie de cualquier función f(x)

$$f\left(x+\Delta x\right)=f\left(x\right)+f'\left(x\right)\Delta x+f''\left(x\right)\frac{\Delta x^2}{2!}+O\left(\Delta x^3\right) \qquad \text{(NR-1)}$$

Si consideramos en el proceso iterativo las aproximaciones x_i y x_{i+1} podemos escribir

$$x_{i+1}=x_i+\Delta x$$

y

$$f\left(x_{i+1}\right)=f\left(x_i\right)+f'\left(x_i\right)\left(x_{i+1}-x_i\right)+f''\left(x_i\right)\frac{\left(x_{i+1}-x_i\right)^2}{2!}+O\left(\Delta x^3\right) \qquad \text{(NR-2)}$$

Análisis del error en el método de Newton-Raphson

El método de Newton-Raphson se obtiene al escribir

$$f\left(x_{i+1}\right) \approx f\left(x_i\right) + f'\left(x_i\right)\left(x_{i+1} - x_i\right) \qquad \text{(NR-3)}$$

y al suponer que x_{i+1} es la raíz llegamos a

$$0 \approx f\left(x_i\right) + f'\left(x_i\right)\left(x_{i+1} - x_i\right)$$

$$x_{i+1} = x_i - \frac{f\left(x_i\right)}{f'\left(x_i\right)} \qquad \text{(NR-4)}$$

Para obtener un estimado del error que se comete con esta aproximación, tenemos que del teorema del residuo, podemos escribir que existe ξ en $[x_i, x_{i+1}]$ tal que

$$f\left(x_{i+1}\right) = f\left(x_i\right) + f'\left(x_i\right)\left(x_{i+1} - x_i\right) + f''\left(x_i\right)\frac{\left(x_{i+1} - x_i\right)^2}{2!} + \ldots + f^{(n)}\left(\xi\right)\frac{\left(x_{i+1} - x_i\right)^n}{n!}$$

Análisis del error en el método de Newton-Raphson

Luego, podemos escribir, hasta orden 2

$$f\left(x_{i+1}\right)=f\left(x_i\right)+f'\left(x_i\right)\left(x_{i+1}-x_i\right)+f''\left(\xi\right)\frac{\left(x_{i+1}-x_i\right)^2}{2!} \qquad \text{(NR-5)}$$

Si x_{i+1} es la raíz x_r tendremos en (NR-5)

$$0=f\left(x_i\right)+f'\left(x_i\right)\left(x_r-x_i\right)+f''\left(\xi\right)\frac{\left(x_r-x_i\right)^2}{2!} \qquad \text{(NR-6)}$$

Luego, si escribimos el error verdadero en el paso i como

$$E_i=E_{t,i}=x_r-x_i$$

tendremos

$$0=f\left(x_i\right)+f'\left(x_i\right)\left(x_{i+1}-x_i\right)=f\left(x_i\right)+f'\left(x_i\right)\left(-x_r+x_r+x_{i+1}-x_i\right)$$

$$0=f\left(x_i\right)+f'\left(x_i\right)\left[-x_r+x_{i+1}+x_r-x_i\right]=f\left(x_i\right)+f'\left(x_i\right)\left[E_{t,i}-E_{t,i+1}\right] \qquad \text{(NR-7)}$$

Análisis del error en el método de Newton-Raphson

Restándole (NR-7) a (NR-6) obtenemos

$$0 = f(x_i) + f'(x_i)(x_r - x_i) + f''(\xi)\frac{(x_r - x_i)^2}{2!} - \left\{ f(x_i) + f'(x_i)\left[E_{t,i} - E_{t,i+1}\right]\right\}$$

Simplificando e introduciendo la definición del error verdadero

$$0 = f'(x_i)\left(E_{t,i}\right) + f''(\xi)\frac{\left(E_{t,i}\right)^2}{2!} - f'(x_i)\left[E_{t,i} - E_{t,i+1}\right]$$

$$0 = f''(\xi)\frac{\left(E_{t,i}\right)^2}{2!} + f'(x_i)E_{t,i+1}$$

Despejando

$$E_{t,i+1} = -\frac{f''(\xi)}{2f'(x_i)}\left(E_{t,i}\right)^2 \qquad \text{(NR-8)}$$

Análisis del error en el método de Newton-Raphson

Si hay convergencia entonces

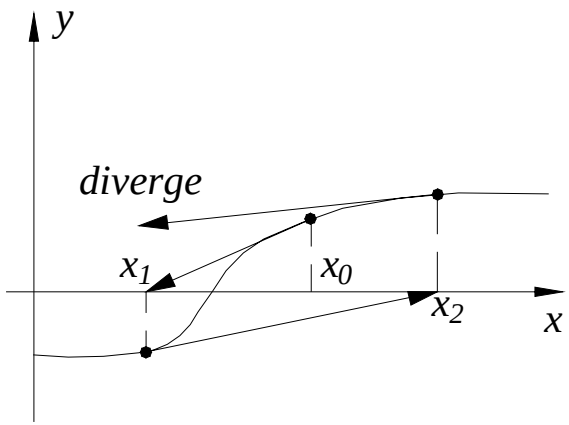
$$\begin{cases} x_i \to x_r \\ \xi \to x_r \end{cases}$$

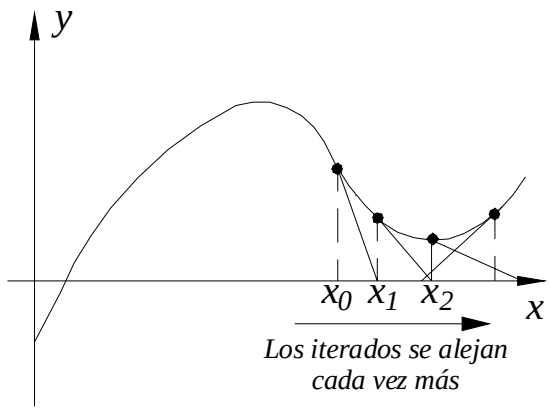
Luego, en (NR-8)

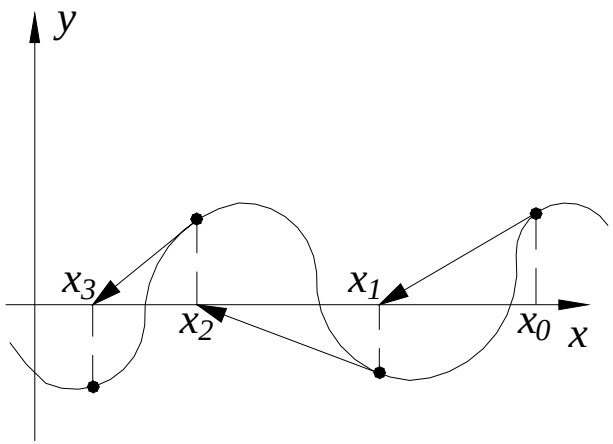
$$E_{t,i+1} = -\frac{f''(x_r)}{2f'(x_r)}\left(E_{t,i}\right)^2 \qquad \text{(NR-9)}$$

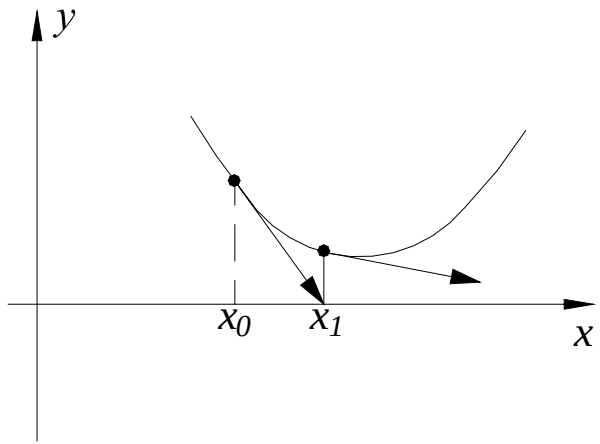
Luego, la convergencia es cuadrática, esto es, el error disminuye proporcionalmente con el cuadrado del error cometido en la iteración anterior.

Casos críticos para método de Newton-Raphson



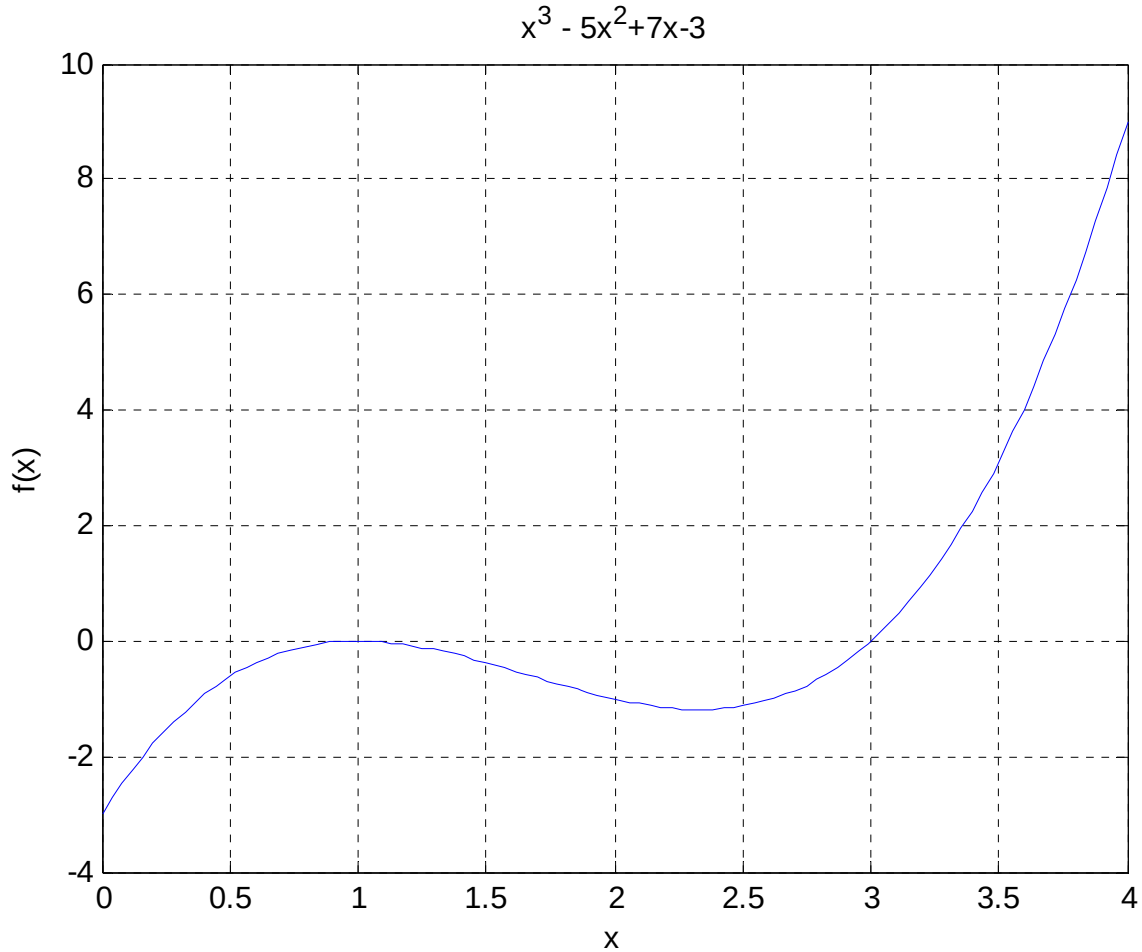






Considere la función

$$f(x) = x^3 - 5x^2 + 7x - 3$$



Halle las raíces utilizando el método de Newton-Raphson partiendo de los siguientes valores: x0=0, 1, 2 y 3, con tolerancias e_tol y desv = 0.0001.

Los resultados se muestran a continuación:

x0	número_iteraciones	raíz
0	14	0.99991
1	1	NaN
2	2	Nan
3	1	3

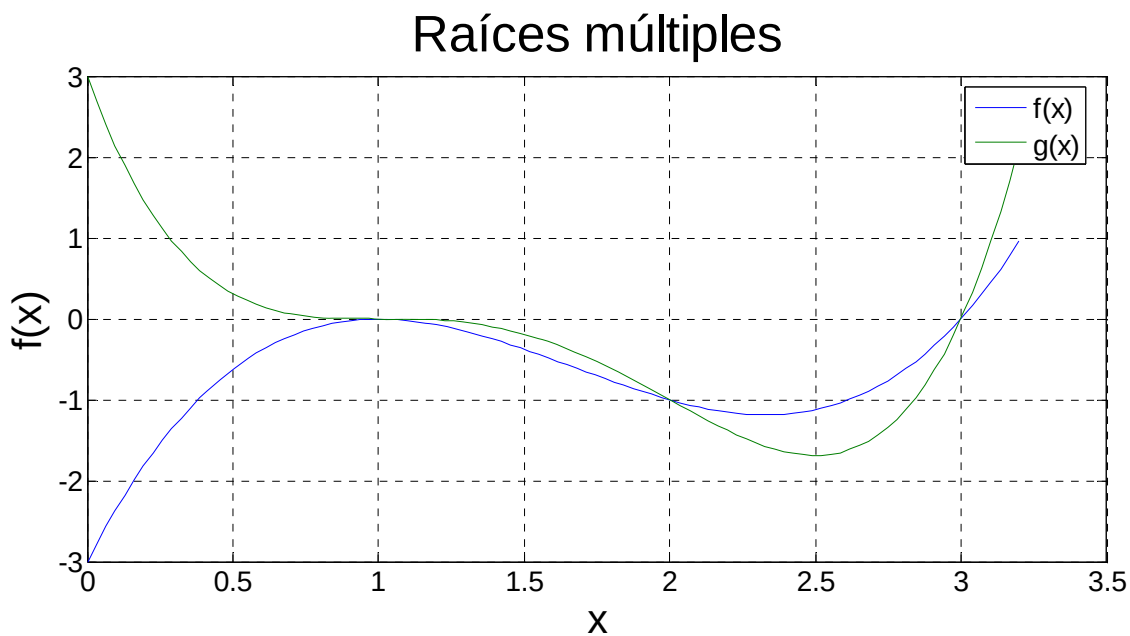
¿A que se deben estos resultados?

Una raíz múltiple a f(x)=0 corresponde a aquella en la cual la función f(x) es tangente al eje x.

Como ejemplo, observe las funciones f y g con raíces múltiples en x=1

$$f(x)=(x-3)(x-1)(x-1)=x^3-5x^2+7x-3$$

$$g(x)=(x-3)(x-1)(x-1)(x-1)=x^4-6x+12x^2-10x+3$$



Nótese que dependiendo de la paridad de la raíz, la función cruzará (multiplicidad impar) o no (multiplicidad par) el eje.

Esto puede verse fácilmente en el caso de polinomios o cuando puede expresarse

$$f(x) = (x - x_r)^n \, h(x)$$

donde x_r es la raíz múltiple y n el índice de multiplicidad.

En este caso

$$si \text{ n es par: } \operatorname{sgn}[f(x)] = \operatorname{sgn}[h(x)] \quad \forall x \text{ alrededor de } x_r$$

$$si \text{ n es impar: } \operatorname{sgn}[f(x)] = \begin{cases} \operatorname{sgn}[h(x)] & \forall x > x_r \\ \operatorname{sgn}[h(x)] & \forall x < x_r \end{cases}$$

Dado que la derivada de f(x) es dada por

$$f'(x) = n\left(x - x_r\right)^{n-1} h(x) + \left(x - x_r\right)^n h'(x)$$

entonces, en $x=x_r$ tendremos que

$$f'(x_r) = 0$$

y, en consecuencia, el método de Newton-Raphson

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

no puede ser usado.

Una variante del método de N-R es dada por

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

dado que la función u(x) tiene los mismos ceros de f(x) como puede verse a partir de

$$u(x) = \frac{f(x)}{f'(x)} = \frac{(x - x_r)^n h(x)}{n(x - x_r)^{n-1} h(x) + (x - x_r)^n h'(x)}$$

$$u(x_r) = \lim_{x \to x_r} \left[u(x) \right] = \frac{(x - x_r) h(x)}{n h(x) + (x - x_r) h'(x)} = 0$$

Este método se denomina método de Newton-Raphson modificado y las iteraciones se realizan a partir de

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

$$u'(x) = \left(\frac{f(x)}{f'(x)} \right)' = \frac{f'(x)\, f'(x) - f''(x)\, f(x)}{\left[f'(x) \right]^2}$$

$$x_{i+1} = x_i - \frac{u(x_i)}{u'(x_i)} = x_i - \left[\frac{\dfrac{f(x)}{f'(x)}}{\dfrac{f'(x)\, f'(x) - f''(x)\, f(x)}{\left[f'(x) \right]^2}} \right]_{x=x_i}$$

$$\boxed{x_{i+1} = x_i - \frac{f(x_i)\, f'(x_i)}{f'^2(x_i) - f''(x_i)\, f(x_i)}}$$

Convergencia del método de N-R para raíces múltiples

El método de Newton-Raphson converge “cuadráticamente” de acuerdo con:

$$E_{t,i+1} = -\frac{f''(x_r)}{2f'(x_r)}\left(E_{t,i}\right)^2$$

Luego, en el caso de utilizarlo en presencia de raíces múltiples tendremos que

$$f'(x) = (x-x_r)^{n-1}\left[nh(x) + (x-x_r)h'(x)\right]$$

$$f''(x) = (x-x_r)^{n-2}\left\{(n-1)\left[nh(x) + (x-x_r)h'(x)\right] + (x-x_r)\left[(n+1)h'(x) + (x-x_r)h''(x)\right]\right\}$$

$$E_{t,i+1} = -\left[\frac{(x-x_r)^{n-2}\left\{(n-1)\left[nh(x) + (x-x_r)h'(x)\right] + (x-x_r)\left[(n+1)h'(x) + (x-x_r)h''(x)\right]\right\}}{2(x-x_r)^{n-1}\left[nh(x) + (x-x_r)h'(x)\right]}\right]_{x\to x_r}\left(E_{t,i}\right)^2$$

Convergencia del método de N-R para raíces múltiples

Simplificando

$$E_{t,i+1} = -\frac{(n-1)}{2(x-x_r)}\left(E_{t,i}\right)^2$$

Pero como

$$E_{t,i} = (x - x_r)$$

Tendremos que la convergencia esta vez es

$$E_{t,i+1} = -\frac{(n-1)}{2} E_{t,i}$$

y, en consecuencia, en el caso de raíces múltiples la convergencia es lineal.

Puesto que

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

Genera una secuencia dada por x_1, x_2, x_3,..,x_n donde a cada término se le agrega

$$-\frac{f(x_i)}{f'(x_i)}$$

una idea para acelerar la convergencia del método consiste en multiplicar el segundo término del lado derecho por ω de manera que

$$x_{i+1} = x_i - \omega\frac{f(x_i)}{f'(x_i)} \qquad 0.5 \le \omega \le 1.5$$

Si ω<1 el método se denomina “subrelajado” mientras que si ω>1 el método se denomina “sobrerelajado (SOR)”

El método de Newton – Raphson sugiere la fórmula de iteración:

$$x_{n+1} = g(x_n) = x_n - \frac{f(x_n)}{f'(x_n)}$$

El método de punto fijo comienza con la ecuación f(x)=0, esta se reorganiza en una expresión equivalente de la forma $x = g(x)$

donde g(x) es algún despeje de "*x*". Luego se crea el proceso iterativo

$$x_{n+1} = g(x_n)$$

Por ejemplo

$$x^2 - 2x + 3 = 0 \Rightarrow x = \frac{x^2 + 3}{2}; \Rightarrow x^{(n+1)} = \frac{x^{(n)^2} + 3}{2}$$

Aplicación: utilizando el método de punto fijo halle la raíz cercana a 4 de

$$x^2 - 2x - 3 = 0$$

Despejando x tenemos $x = \sqrt{2x+3}$

Luego se crea el proceso iterativo $x_{n+1} = g(x_n) \Rightarrow x_{n+1} = \sqrt{2x_n + 3}$

Para obtener

n	x0	g(x0)	x1	f(x3)	Error (%)
0	4	3.31662479	3.31662479	1.36675042	20.60454
1	3.31662479	3.10374767	3.10374767	0.42575425	6.85871
2	3.10374767	3.03438550	3.03438550	0.13872434	2.28587
3	3.03438550	3.01144002	3.01144002	0.04589095	0.76194
4	3.01144002	3.00381092	3.00381092	0.01525820	0.25398
5	3.00381092	3.00127004	3.00127004	0.00508176	0.08466
6	3.00127004	3.00042332	3.00042332	0.00169344	0.02822
7	3.00042332	3.00014110	3.00014110	0.00056443	0.00941
8	3.00014110	3.00004703	3.00004703	0.00018814	0.00314
9	3.00004703	3.00001568	3.00001568	0.00006271	0.00105

¿Qué pasa en el caso anterior si consideramos otros despejes?. Por ejemplo si tomamos

$$x^2 - 2x - 3 = 0 \quad \Rightarrow \quad x(x-2) - 3 = 0 \quad \Rightarrow \quad x = \frac{3}{x-2} \quad \Rightarrow \quad x_{n+1} = \frac{3}{x_n - 2}$$

Para obtener

n	x0	g(x0)	x1	f(x3)	Error (%)
0	4	1.500000	1.50000000	-3.75000000	166.66667
1	1.500000	-6.000000	-6.00000000	45.00000000	125.00000
2	-6.000000	-0.375000	-0.37500000	-2.10937500	1500.00000
3	-0.375000	-1.263158	-1.26315789	1.12188366	70.31250
4	-1.263158	-0.919355	-0.91935484	-0.31607700	37.39612
5	-0.919355	-1.027624	-1.02762431	0.11126034	10.53590
6	-1.027624	-0.990876	-0.99087591	-0.03641310	3.70868
7	-0.990876	-1.003051	-1.00305064	0.01221187	1.21377
8	-1.003051	-0.998984	-0.99898415	-0.00406236	0.40706
9	-0.998984	-1.000339	-1.00033873	0.00135504	0.13541

Converge a otra raíz! (-1). ¡Que raro!

Probemos con otro despeje. Si hacemos

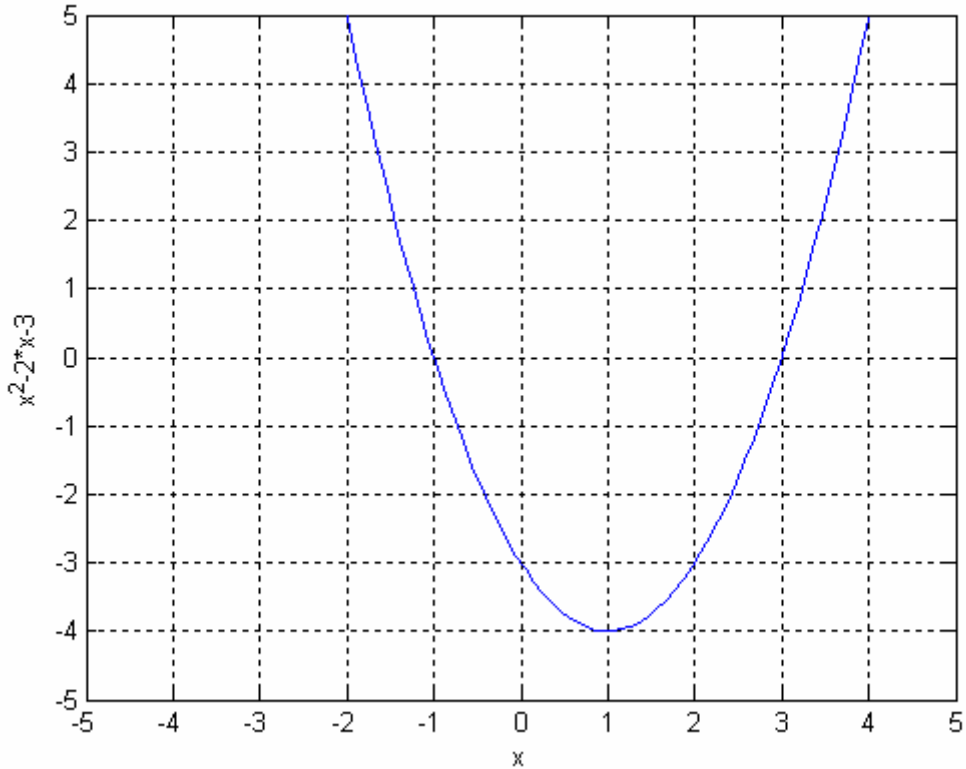
$$x^2 - 2x - 3 = 0 \quad \Rightarrow \quad x^2 - 3 = 2x \quad \Rightarrow \quad x = \frac{x^2 - 3}{2}$$

Para obtener

n	x0	g(x0)	x1	f(x3)	Error (%)
0	4.0000E+00	6.5000E+00	6.5000E+00	2.6250E+01	38.46154
1	6.5000E+00	1.9625E+01	1.9625E+01	3.4289E+02	66.87898
2	1.9625E+01	1.9107E+02	1.9107E+02	3.6123E+04	89.72891
3	1.9107E+02	1.8252E+04	1.8252E+04	3.3311E+08	98.95318
4	1.8252E+04	1.6658E+08	1.6658E+08	2.7747E+16	99.98904
5	1.6658E+08	1.3874E+16	1.3874E+16	1.9248E+32	100.00000
6	1.3874E+16	9.6240E+31	9.6240E+31	9.2622E+63	100.00000
7	9.6240E+31	4.6311E+63	4.6311E+63	2.1447E+127	100.00000

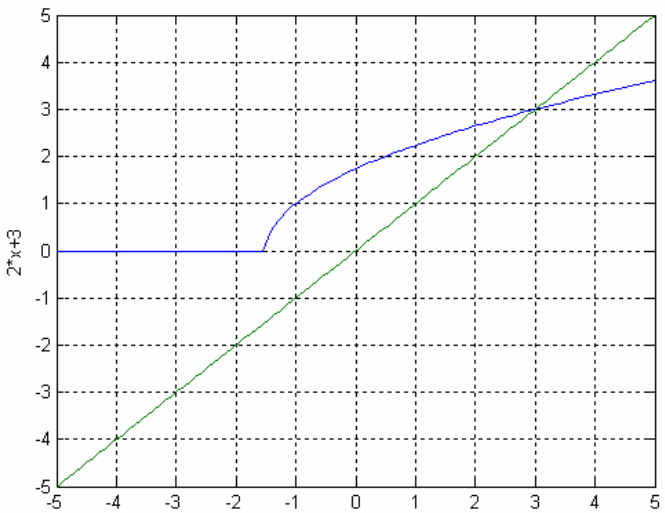
No hay convergencia. ¿Qué está pasando?

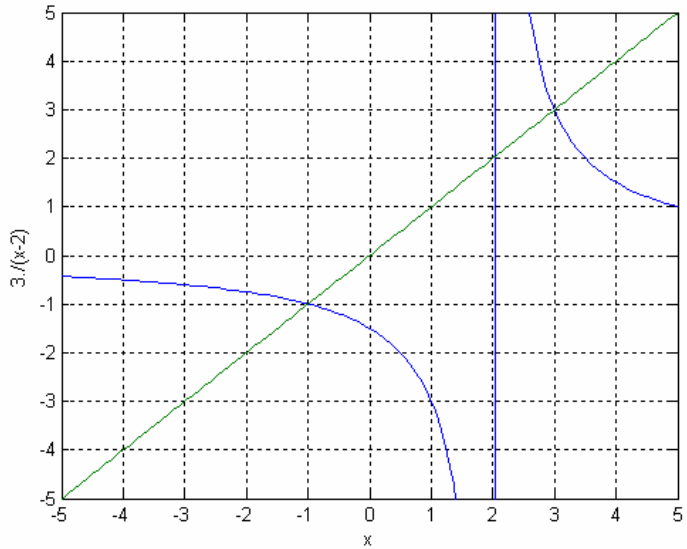
La gráfica de la función $f(x)=x^2-2x-3$

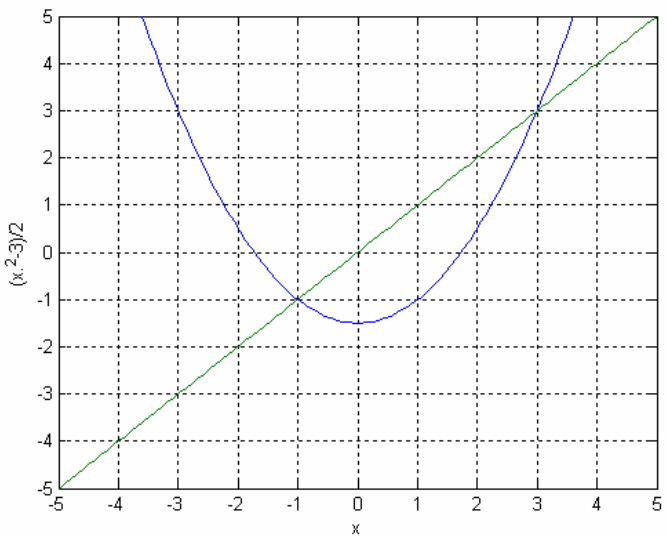


Muestra las raíces en x=-1 y x=3

La función asociada a cada despeje, tienen como gráfica



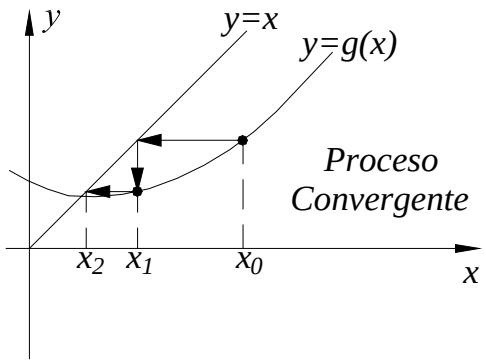


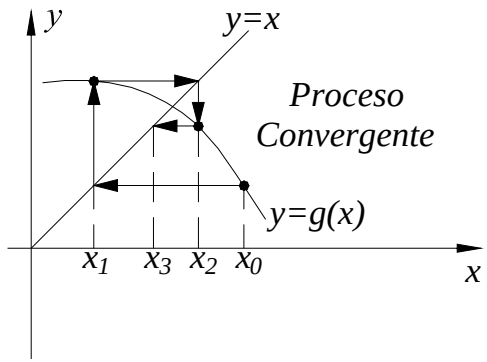


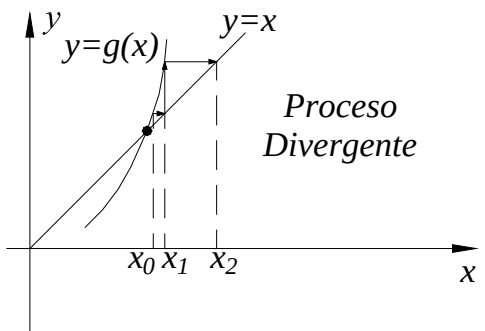
¿Qué diferencia existe entre las distintas curvas aquí mostradas para que haya convergencia sólo en algunos casos?

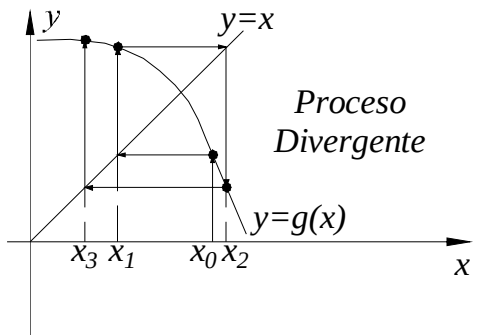
Dificultades del Método

- Existen en general distintas opciones de despejes, usualmente despejes distintos convergen a raíces diferentes.
- El método presenta problemas de convergencia.









En general, si g(x) y g´(x) son continuas en un intervalo alrededor de la raíz *r* de la ecuación x=g(x) y sí |g´(x)|<1 para todas las x en el intervalo, entonces $x_{n+1}=g(x_n)$, n=1, 2, 3... convergerá a la raíz siempre que la semilla de iteración éste en el intervalo.

Convergencia del método de Punto Fijo

El método de punto fijo se basa en la construcción de un proceso iterativo en el cual del problema hallar x que satisface

$$f(x) = 0 \qquad \text{(PF-1)}$$

pasamos a resolver

$$x = g(x) \qquad \text{(PF-2)}$$

a partir del proceso iterativo

$$x_{i+1} = g(x_i) \qquad \text{(PF-3)}$$

Sí la solución es x_r tal que

$$x_r = g(x_r) \qquad \text{(PF-4)}$$

Restando (PF-3) a (PF-4) tenemos

$$x_r - x_{i+1} = g(x_r) - g(x_i) \qquad \text{(PF-5)}$$

El teorema del valor medio de la derivada establece que si g(x) y g'(x) son funciones continuas en [a,b] entonces existe x=ξ en [a,b] tal que

$$g'(\xi) = \frac{g(b) - g(a)}{b - a} \qquad \text{(PF-6)}$$

Luego, en [x_i,x_r] existe ξ tal que

$$g'(\xi) = \frac{g(x_r) - g(x_i)}{x_r - x_i} \qquad \text{(PF-7)}$$

Convergencia del método de Punto Fijo

Despejando tenemos

$$g(x_r) - g(x_i) = g'(\xi)(x_r - x_i) \qquad \text{(PF-8)}$$

Sustituyendo (PF-8) en (PF-5) obtenemos

$$x_r - x_{i+1} = g'(\xi)(x_r - x_i) \qquad \text{(PF-9)}$$

Puesto que el error verdadero viene dado por

$$E_{t,i} = x_r - x_{i+1} \qquad \text{(PF-10)}$$

Luego, dado que para convergencia requerimos que

$$\left| \frac{E_{t,i+1}}{E_{t,i}} \right| < 1 \qquad \text{(PF-11)}$$

Convergencia del método de Punto Fijo

entonces

$$\left|\frac{E_{t,i+1}}{E_{t,i}}\right| = \left|\frac{x_r - x_{i+1}}{x_r - x_i}\right| = \left|g'(\xi)\right| < 1 \qquad \text{(PF-12)}$$

y, en consecuencia

$$\left|E_{t,i+1}\right| = \left|E_{t,i}\right|\left|g'(\xi)\right| \qquad \text{(PF-13)}$$

Para convergencia x_r, es cercano a ξ. Luego

$$\left|E_{t,i+1}\right| = \left|E_{t,i}\right|\left|g'(x_r)\right| \qquad \text{(PF-14)}$$

Luego, (PF-14) nos indica que el error disminuye de manera lineal en el método de punto fijo. El factor de proporcionalidad es el modulo de g'(xr) el cual debe cumplir con $\left|g'(x_r)\right| < 1$

Convergencia acelerada: Método de Aitken

La convergencia del método de punto fijo es lineal. Esto es, la sucesión

$$\{p_n\}_{n=0}^{\infty} \to p \Rightarrow (p_{n+1} - p) \approx \varepsilon (p_n - p) \quad \text{para} \quad n >> 1$$

Luego, si estamos relativamente cerca de p, las diferencias

$$(p_n - p), (p_{n+1} - p), (p_{n+2} - p)$$

tienen el mismo signo (irrelevante como se discutirá más adelante) entonces

$$\frac{p_{n+1} - p}{p_n - p} \approx \frac{p_{n+2} - p}{p_{n+1} - p}$$

Luego

$$(p_{n+1} - p)^2 \approx (p_n - p)(p_{n+2} - p)$$

Convergencia acelerada: Método de Aitken

Desarrollando la última expresión llegamos a

$$p = \frac{p_n p_{n+2} - {p_{n+1}}^2}{p_n + p_{n+2} - 2p_{n+1}}$$

que puede escribirse como

$$p = \frac{p_n p_{n+2} - \left(p_{n+1} - p_n\right)^2 + {p_n}^2 - 2p_{n+1}p_n}{p_n + p_{n+2} - 2p_{n+1}}$$

Reagrupando los términos del numerador

$$p = \frac{p_n \left(p_{n+2} + p_n - 2p_{n+1}\right) - \left(p_{n+1} - p_n\right)^2}{p_n + p_{n+2} - 2p_{n+1}}$$

Simplificando llegamos a

$$p = p_n - \frac{\left(p_{n+1} - p_n\right)^2}{p_n + p_{n+2} - 2p_{n+1}}$$

Convergencia acelerada: Método de Aitken

El método de Aitken se basa en que la sucesión

$$\{p_n\}_{n=0}^{\infty}$$

definida por

$$\hat{p}_n = p_n - \frac{(p_{n+1} - p_n)^2}{p_n + p_{n+2} - 2p_{n+1}}$$

converge más rápidamente a p que la sucesión original.

Método de Steffensen

La aplicación del método de Aitken se conoce como método de Steffensen.

Los pasos a seguir son:

1. Escoger $p_0 = p_0^{(0)}$

2.Calcular $p_1 = p_1^{(0)} = g\left(p_0^{(0)}\right)$

3.Calcular $p_2 = p_2^{(0)} = g\left(p_1^{(0)}\right)$

4.Calcular $\hat{p}_0 = p_0 - \dfrac{\left(p_1 - p_0\right)^2}{p_0 + p_2 - 2p_1}$

5.Hacer $p_0^{(1)} = \hat{p}_0$ y repetir 1-4 hasta convergencia

Cuidado con el denominador, que puede hacerse nulo. Si el denominador es muy pequeño, parar y quedarse con la última aproximación.

Método de Steffensen

Para encontrar una solución a $p = g(p)$ dada una aproximación inicial p_0:

ENTRADA aproximación inicial p_0; tolerancia TOL; número máximo de iteraciones N_0.

SALIDA solución aproximada p o mensaje de falla.

Paso 1 Tome $i = 1$;

Paso 2 Mientras $i \leq N_0$ haga pasos 3-6.

Paso 3 Tome $p_1 = g(p_0)$; (*Calcule* $p_1^{(i-1)}$.)
$p_2 = g(p_1)$; (*Calcule* $p_2^{(i-1)}$.)
$p = p_0 - (p_1 - p_0)^2/(p_2 - 2p_1 + p_0)$.(*Calcule* $p_0^{(i)}$.)

Paso 4 Si $|p - p_0| < TOL$ entonces
SALIDA (p); (*Procedimiento terminado satisfactoriamente.*)
PARE.

Paso 5 Tome $i = i + 1$.

Paso 6 Tome $p_0 = p$. (*Redefina* p_0.)

Paso 7 SALIDA ('El método falló después de N_0 iteraciones, $N_0 =$ ', N_0);
(*Procedimiento terminado sin éxito.*)
PARE.

Método de Steffensen

```
1   function p = steffensen(p0,TOL,N0)
2   % Encuentra una aproximación a p=g(p) dada una aproximación p0
3   % Entrada:       p0 :    Aproximación inicial
4   %                TOL:    Tolerancia o Error local
5   %                N0 :    Numéro máximo de iteraciones
6   % Salida :       p : solucíon aproximada
7   %                o
8   %                Mensaje indicando falla del método
9
10  % Definición de la función g
11
12  g = inline('1/2-exp(-x/(2*pi))*sin(x)');
13  i=1;
14  while (i<=N0)
15      p1 = g(p0);
16      p2 = g(p1);
17      p = p0 - ((p1-p0)^2)/(p2-2*p1+p0)
18      if abs(p-p0)<TOL
19          disp([' p = ' num2str(p)]);
20          disp(['Solución alcanzada en ',num2str(i),' iteraciones']);
21          return
22      end
23      i=i+1;
24      p0=p;
25  end
26  disp(['El método falló después de ',num2str(N0),' iteraciones']);
```

Aceleración de Aitken

Consideremos las secuencias generadas por el método de punto fijo y de Aitken en la función

$$f(x) = \frac{1}{2} - e^{-\frac{x}{2\pi}} sen(x) - x$$

utilizando el despeje $x = g(x) = \frac{1}{2} - e^{-\frac{x}{2\pi}} sen(x)$

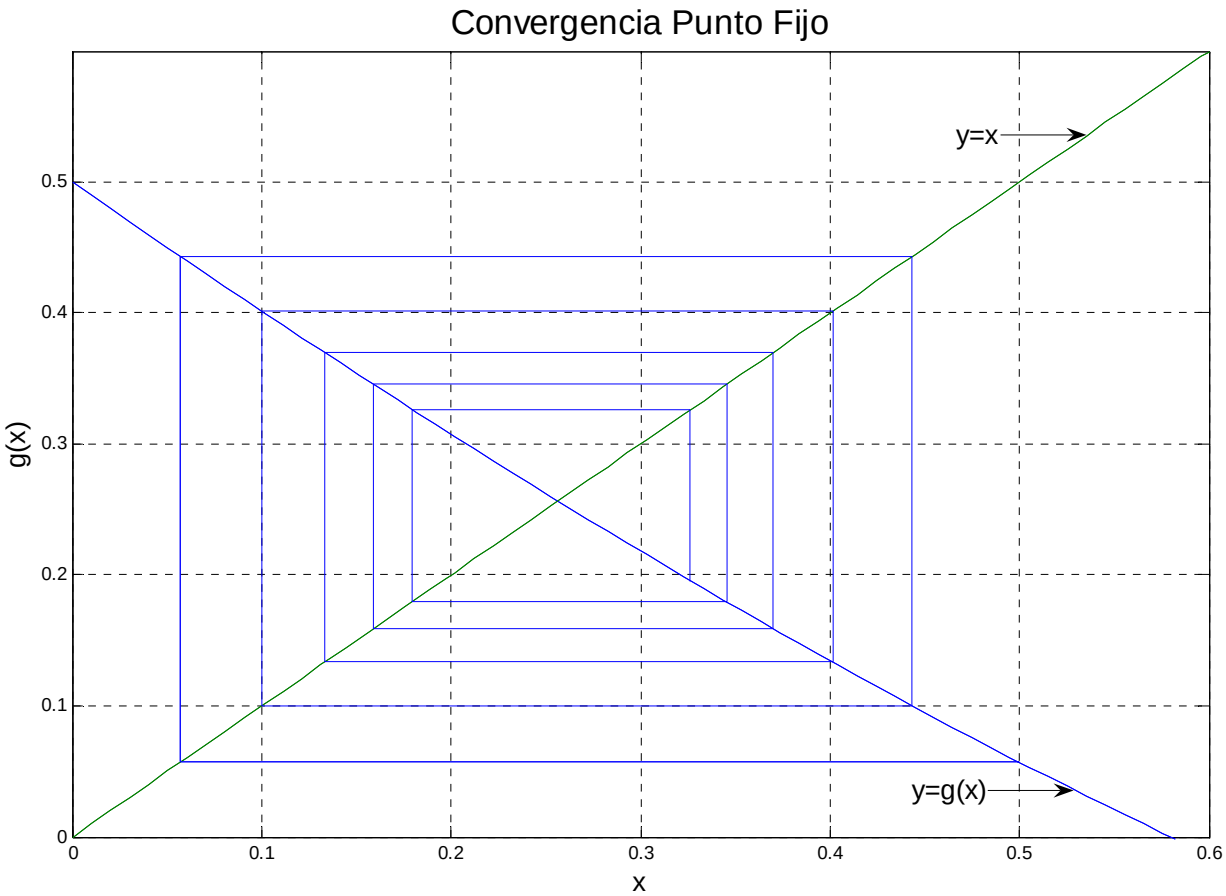
Obtenemos, con p_0 = 0.5 y tolerancia 0.00000000001 en 4 iteraciones

i	p0	p1	p2	p
1	0.500000000	0.057247412	0.443302792	0.263479766687873 93
2	0.263479767	0.250253772	0.262019993	0.256480510039524 66
3	0.256480510	0.256468892	0.256479230	0.256474362626799 85
4	0.256474363	0.256474363	0.256474363	0.256474362622058 31

¡El método de punto fijo “clásico” requiere de 211 iteraciones en este mismo caso!

Aceleración de Aitken

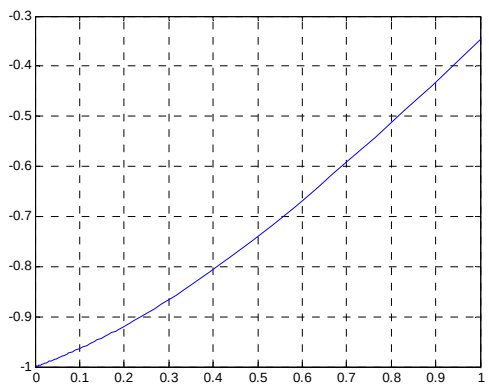
La gráfica siguiente muestra como ocurre la convergencia al utilizar el método de punto fijo en este caso



i	p0	p
1	0.500000000	0.05724741238968151
2	0.057247412	0.44330279224377950
3	0.443302792	0.10029409146178031
4	0.100294091	0.40145951908104061
5	0.401459519	0.13342431453987363
6	0.133424315	0.36976630700916019
7	0.369766307	0.15925702773992101
8	0.159257028	0.34538437975725400
9	0.345384380	0.17954978763845725
10	0.179549788	0.32644448924384140

Este comportamiento es fácilmente entendido si se calcula la derivada de la función g(x) (el “despeje” que se hizo de f(x)=0). Para ello utilizamos MATLAB:

```
>> syms x
>> diff(1/2-exp(-x/(2*pi))*sin(x))
 ans =
 1/2/pi*exp(-1/2*x/pi)*sin(x)-exp(-1/2*x/pi)*cos(x)
 >> g_prima=inline('1/2/pi*exp(-1/2*x/pi)*sin(x)-exp(-1/2*x/pi)*cos(x)');
>> fplot(g_prima,[0 1])
>> grid on
```



Nótese que los valores de g’(x) son negativos en [0,1] pero con | g’(x) |<1, lo que explica la forma en la que converge punto fijo en este caso.

MATLAB posee comandos especialmente diseñados para calcular raíces de polinomios. Por ejemplo, para hallar las raíces del polinomio

$$f(x) = x^3 + x^2 - 3x - 3 = 0$$

se introduce el comando

```
>> roots([1,1,-3,-3])

ans =

    1.7321

   -1.7321

   -1.0000
```

Para una función usamos fzero. Veamos. Si

$$f\left(x\right)=x^2\sin(x)-2=0$$

```
>> fzero('(x^2)*sin(x)-2',1)

ans =

    1.4221
```

Sin embargo, podemos tener algunas decepciones. Por ejemplo si hacemos

```
>> fzero('x^2',1)
```

obtenemos

```
Exiting fzero: aborting search for an interval containing a
sign change because NaN or Inf function value
encountered during search.

(Function value at -1.7162e+154 is Inf.)

Check function or try again with a different starting value.

ans =

   NaN
```

Luego, un conocimiento de lo que se está haciendo es sumamente importante para lograr una aplicación exitosa de los métodos numéricos.

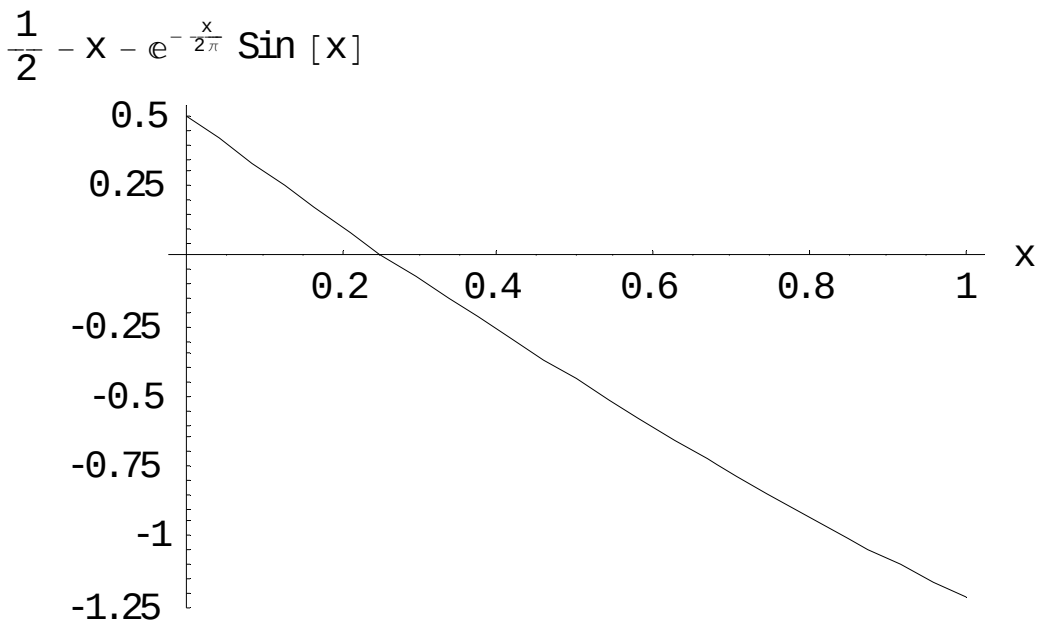
In[1]:=
$$f[x\_] := \frac{1}{2} - \text{Exp}\left[-\frac{x}{2\pi}\right] \text{Sin}[x] - x$$

In[3]:=
```
FindRoot[f[x] == 0, {x, 0.5}]
```

Out[3]=
$$\{x \to 0.256474\}$$

In[4]:=
```
Plot[f[x], {x, 0, 1}, AxesLabel → {x, f[x]}];
```

From In[4]:=



“La libertad de los muchos, perezosos o seducidos por la tiranía, se salva casi siempre por la determinación indomable de unos pocos que pelean contra lo que parece irremediable, contra lo verosímil predicado por los acomodaticios, contra lo que la prudencia sobornada por el dominio aconseja como más recomendable”

Fernando Sabater

MECÁNICA COMPUTACIONAL I

Capítulo 2

Solución de ecuaciones no –lineales