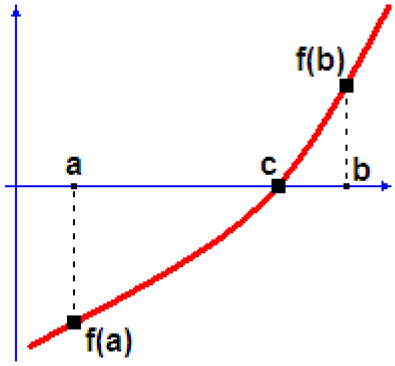
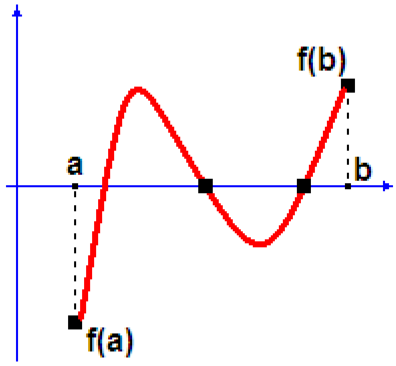
# Práctica 2 – Máxima – Resolución numérica de Ecuaciones

**<u>1.- Teorema de Bolzano:</u>** Sea **f(x) continua en [a,b],** con **f(a)·f(b) < 0** (signos distintos)
Entonces EXISTE AL MENOS un $c \in [a,b]$ t.q. f(c)=0




Es decir, f(x) tiene al menos un cero en ese intervalo, aunque puede tener varios
<u>Nota</u>: Si $f'(x) \leq 0$ en [a,b], ó $f'(x) \geq 0$ en [a,b] entonces **ESE CERO ES ÚNICO en [a,b]**.

## <u>2.-Metodo numéricos para la resolución de ecuaciones.</u>

Se trata de resolver la ecuación f(x)=0, lo cual puede ser complicado analíticamente o imposible.
Por ejemplo, con: $f(x)=3x^5-4x^4+2x^3-x^2+5x-7$ ó $f(x)=e^x-2\cdot x$

<u>Paso 1</u>: Estableceremos intervalos [a,b] que contengan una única raíz de f(x)
<u>Paso 2</u>: Utilizaremos algún método numérico iterativo de forma que $\mathbf{x_k}$ será un aproximación a la solución exacta tras **k** iteraciones.

## <u>3.- Método de bisección.</u>

Sea $f : [a; b] \to R$ de clase $C^0$ t. q. $f(a)\cdot f(b) < 0$, por Bolzano sabemos que $\exists c \in (a,b)$ t.q. f(c)=0

- Llamemos $[a_0,b_0]$ a nuestro intervalo [a,b]
- El error al tomar el punto medio como solución es $\boldsymbol{error} \leq \frac{b-a}{2}$
- Si el error es mayor que el admitido haremos una iteración
  - Llamemos **m** al punto medio de $\mathbf{a_0}$ y $\mathbf{b_0}$ = **m** = $\frac{a_0+b_0}{2}$
  - Si f(m)=0, casualmente lo hemos encontrado: SOL = m. PARAMOS
  - Si $f(m) \neq 0$ elegimos entre $\begin{cases} [a_0,m] \text{ si es que } f(a_0)\cdot f(m)<0, \text{ haciendo } [a_1,b_1]=[a_0,m] \\ [m,b_0] \text{ si es que } f(m)\cdot f(b_0)<0, \text{ haciendo } [a_1,b_1]=[m,b_0] \end{cases}$
  - Calculo de nuevo el error con el nuevo intervalo
  - Si error > $error_{admitido}$
- SOL=punto medio el ultimo intervalo considerado

El error tras **n** iteraciones y tomar como solución el punto medio del último intervalo es $\mathbf{error} \leq \frac{b-a}{2^{n+1}}$
Podemos saber el nº de iteraciones a priori, despejando el primer $n \in N$ que cumpla:

$error < error_{admitido}$, es decir, $\frac{b-a}{2^{n+1}} < error_{admitido}$

Ejemplo:
1º.- Crear la función bisección:

```
biseccion(f,a,b,erroradmitido):=block
  (
    [i, n, m],
    if subst(a,x,f)*subst(b,x,f)>0 then
      (
        print("El intervalo [",a,",",b,"] no cumple Bolzano")
      )
    else
      (
        n: ceiling(log((b-a)/erroradmitido)/log(2)-1),
        print("Se necesitan ",n," iteraciones"),
        for i:1 thru n step 1 do
          (
            print("Iteracion ",i),
            m: (a+b)/2,
            if subst(m,x,f)=0 then
              (
                print("Casualm se halló el valor exacto en ",i," iterac. "),
                i: n
              )
            else
              if subst(a,x,f)*subst(m,x,f)<0 then b: m else a: m
          ),
        print("Valor aprox. tras ",n," iteraciones"),
        float(a+b)/2
      )
  );
```

2º:- Para hallar un cero de $f(x)=x^2-2$ que esté en el intervalo [1,3] y con un error máximo = 0'000001 haré:

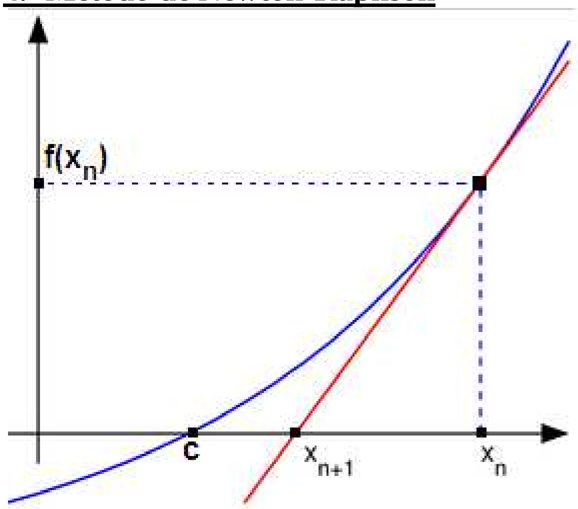
`biseccion(x^2-2, 1, 3, 0.000001)`

Previamente mostrará cuantas iteraciones son necesarias.

**Nota1**: Si casualmente halla una solución exacta parará de iterar y nos avisará.
**Nota2**: Si introducimos un intervalo que no cumple Bolzano también nos avisará y no calculará.

NOTA.GENERAL: Si cumple Bolzano, el método de bisección converge siempre, pero lentamente. (necesito muchas iteraciones).

## 4.- Método de Newton-Raphson




- Sea $f \in \boxed{C^{2)}[a,b]}$, con $\boxed{f(a)\cdot f(b) < 0}$ , $\boxed{f'' \text{ signo cte en } [a,b]}$
- Partimos de una aproximación $\mathbf{x_0}$ cercana al valor exacto **c**, que cumpla $\boxed{\mathbf{f(x_0)\cdot f''(x_0) > 0}}$
- Calcularemos la recta tangente en el pto $(x_n,f(x_n))$
- El punto de corte de la tangente con el eje de abcisas, $\mathbf{x_{n+1}}$, será una aproximación a **c** mejor que $x_n$.

$$f'(x_n) = m = \frac{f(x_n)-0}{x_n-x_{n+1}}$$

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

Cota del error: Buscamos **m** $\leq |f'(x)|$, **M** $\geq |f''(x)|$ $\forall x \in [a,b]$,

(lo mejor es hacer la gráfica de $|f'(x)|$ y de $|f''(x)|$ y elegir m y M)

m >0 es más pequeño que el mínimo de $f'(x)$ en el intervalo

M>0 es más grande que el mínimo de $f''(x)$ en el intervalo

Hay dos formas de medir el error tras la iteración k

Elijo

a) **error$_k$** < $\dfrac{|f(x_k)|}{m}$, *(esta cota del error es más restrictiva que la siguiente)*

b) error$_k$ < $\dfrac{M}{2\cdot m} \cdot (x_k - x_{k-1})^2$ (la condición b) **suele simplificarse** y paramos de iterar cuando $|x_k - x_{k-1}| < \varepsilon$ prefijado)

**Ejemplo: Hallar los ceros de f(x)=$x^2$-2.**

```
f: x^2-2$
```

*Probaré en el intervalo [1,3]*

```
subst(1,x,f)* subst(3,x,f);
```

*(%o2) -7* *Como es negativo cumple con Bolzano*

```
fp: diff(f,x);
```

*(%o3) 2x*

```
fpp: diff(f,x,2);
```

*(%o4) 2*

Veo que $f''(x)$ tiene el mismo signo en todo el intervalo, si no lo veo a ojo, hago la gráfica de $f''(x)$

*Ahora busco un candidato para $x_0$. Pruebo con $x_0$=1'25*

```
x0:1.25$
subst(x0,x,f)*subst(x0,x,fpp);
```

*(%o6) -0.875*

*No cumple las condiciones para aplicar Newton. Probaré con $x_0$=2*

```
x0:2$
subst(x0,x,f)*subst(x0,x,fpp);
```

*(%o8) 4*

*$x_0$=2 sí que me sirve.*

*Busco un valor para* **m** *(como es creciente en todo el intervalo tomo un valor menor que $|f'(1)|=2$*

```
m:0.9
```

Si hubiera sido más difícil, dibujaría $|f'(x)|$ en el intervalo [1,3] y elegiría **m**

Ahora iterar por ejemplo hasta que error < 0'000001 =$10^{-6}$ ó hasta que $\varepsilon=|x_k-x_{k-1}|<0.001$

```
x1: float( x0-subst(x0,x,f)/subst(x0,x,fp) ); /* comentario */
mierror:abs(subst(x1,x,f))/m;
eps:abs(x1-x0);
```

*(%o64) 1.5* — $x_1$
*(%o65) 0.27777777777778* — *mierror*
*(%o66) 0.5* — $\varepsilon$

```
x2: float( x1-subst(x1,x,f)/subst(x1,x,fp) ); /* comentario */
mierror:abs(subst(x2,x,f))/m;
eps:abs(x2-x1);
```

*(%o67) 1.416666666666667* — $x_2$
*(%o68) 0.0077160493827163* — *mierror*
*(%o69) 0.083333333333333* — $\varepsilon$

```
x3: float( x2-subst(x2,x,f)/subst(x2,x,fp) ); /* kjhgk*/
mierror:abs(subst(x3,x,f))/m;
eps:abs(x3-x2);
```

*(%o70) 1.41421568627451* — $x_3$
*(%o71) 6.6747832031902963·$10^{-6}$* — *mierror*
*(%o72) 0.0024509803921569* — $\varepsilon$

CONCLUSIONES:

- Nos cuesta más elegir y comprobar el intervalo
- Nos ha costado elegir el $x_0$ inicial cercano a la solución exacta c.
- Pero converge muy rápido, converge cuadráticamente
  ($mierror_1$=0'2, $mierror_2$=0'007, $mierror_3$=0'000006)
- Podríamos hacer una función que iterase automáticamente cuando sepáis programar más.

**5.- Ejercicios**

5.1.- Resolver: $\mathbf{x=e^{-x}}$, con 4 decimales exactos

a) Halla previamente un intervalo donde esté garantizado que haya una única solución (no vale con una gráfica)
b) Antes de utilizar el método de bisección calcula cuantas iteraciones son necesarias
c) Calcúlalo con el método de bisección y compara el nº de iteraciones con el apartado b
d) Ahora con el método de Newton-Raphson.
e) Indica las diferencias y problemas encontrados al aplicar un método y otro.

5.2.- Halla el punto de corte de $\mathbf{y=x^2}$ con $\mathbf{y=e^x}$ Debes decirme: *el punto de corte es el* (….., …..)

5.3.- a) Demostrar que la ecuación $x = \tan(x)$ posee infinitas raíces reales.
b) Utilizar el método de bisección para aproximar las dos 1as raíces positivas (6 decimales)

5.4.- Utiliza el mét. de bisección para hallar $\sqrt[3]{17}$ (*observa cómo he hallado* $\sqrt{2}$ *en los ejemplos*)

5.5.- Encontrar la solución positiva de $2\cdot\sin(x) = x$ con los dos métodos.

5.6.- La función $f(x) = \frac{4\cdot x-7}{x-2}$ se hace cero en x = 7/4.
Utilizar el método de Newton para resolver f(x) = 0 con las aproximaciones iniciales:
a) $x_0$ = 1'6; b) $x_0$ = 1'5; c) $x_0$ = 3 Explicar qué ocurre en cada caso.

5.7.- Calcular 4 iteraciones del método de Newton si $f(x) = x^{1/3}$ tomando $x_0$ = 0'1. ¿Qué ocurre?

5.8.- a) Halla un intervalo donde encuentres un cero de $f(x) = x^3+x-3$
b) Demuestra que es ese cero es único. Hállalo con los 2 métodos