Tema 2 - Operaciones matemáticas básicas

Cálculo de raíces

M. en C. Gustavo Contreras Mayén

18 de septiembre de 2014

Contenido

Contenido

Contenido

Contenido

Contenido

Contenido

Contenido

Contenido

Contenido

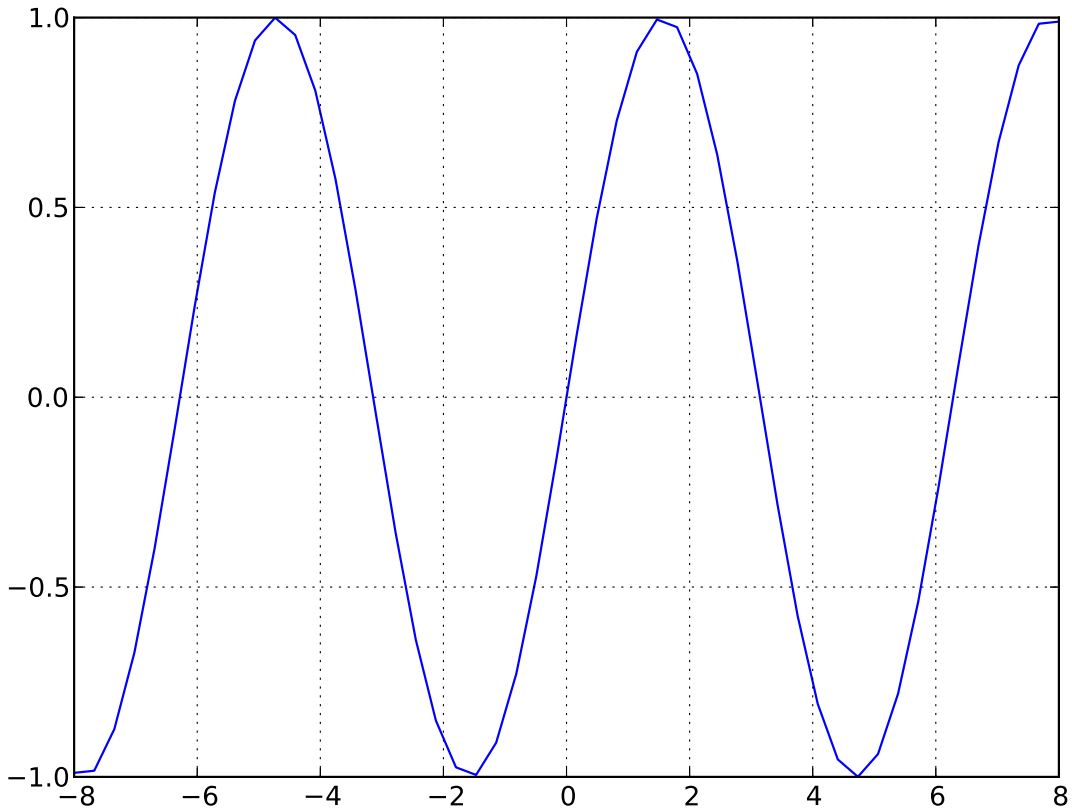
Contenido

Cálculo de raíces

Sea $y = f(x)$. Los valores de x que hacen que $y = 0$ se denominan raíces de la ecuación.

El teorema fundamental del álgebra indica que todo polinomio de grado n, tiene n raíces. En el caso de las raíces reales, se tiene que corresponden a los valores x que hacen que la función corte el eje de las abscisas:

Ejemplo de la función seno(x)



Las raíces de un polinomio pueden ser reales o complejas.

Si un polinomio tiene coeficientes reales

$$a_0, a_1, a_2, \ldots, a_{n-1}, a_n$$

entonces todas las raíces complejas siempre ocurrirán en pares conjugados complejos.

Por ejemplo, un polinomio cúbico tiene la siguiente forma general:

$$f(x) = a_0x^3 + a_1x^2 + a_2x + a_3$$

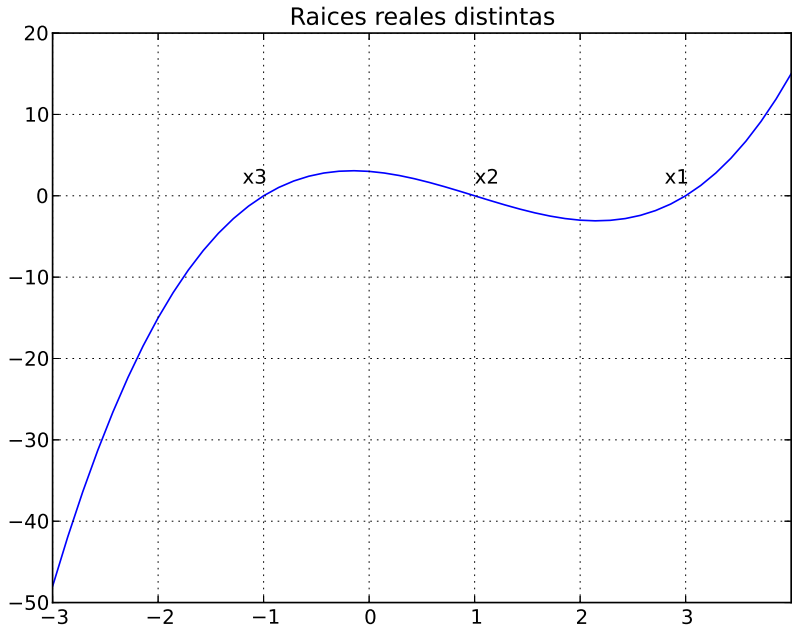
1. Tres raíces reales distintas.
2. Una raíz real con multiplicidad 3.
3. Una raíz real simple y una raíz real con multiplicidad 2.
4. Una raíz real y un par conjugado complejo.

Tres raíces distintas

$$\begin{aligned} f(x) =& x^3 - 3x^2 - x + 3 \\ =& (x-3)(x+1)(x-1) \end{aligned}$$

Las raíces son:

$$\begin{aligned} x_1 =& 3 \\ x_2 =& -1 \\ x_3 =& 1 \end{aligned}$$

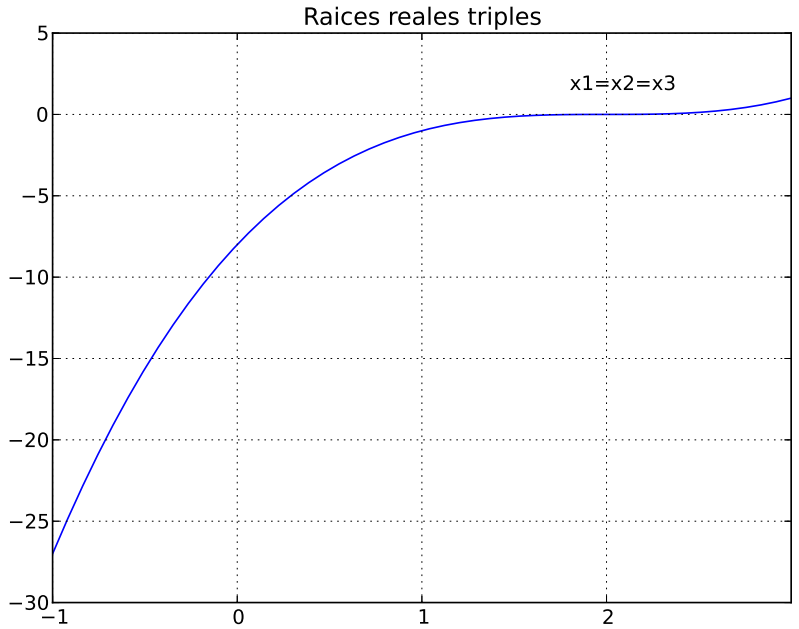


Raíz real con multiplicidad 3

$$\begin{aligned} f(x) =& x^3 - 6x^2 + 12x - 8 \\ =& (x-2)^3 \end{aligned}$$

Las raíces son:

$$\begin{aligned} x_1 =& 2 \\ x_2 =& 2 \\ x_3 =& 2 \end{aligned}$$

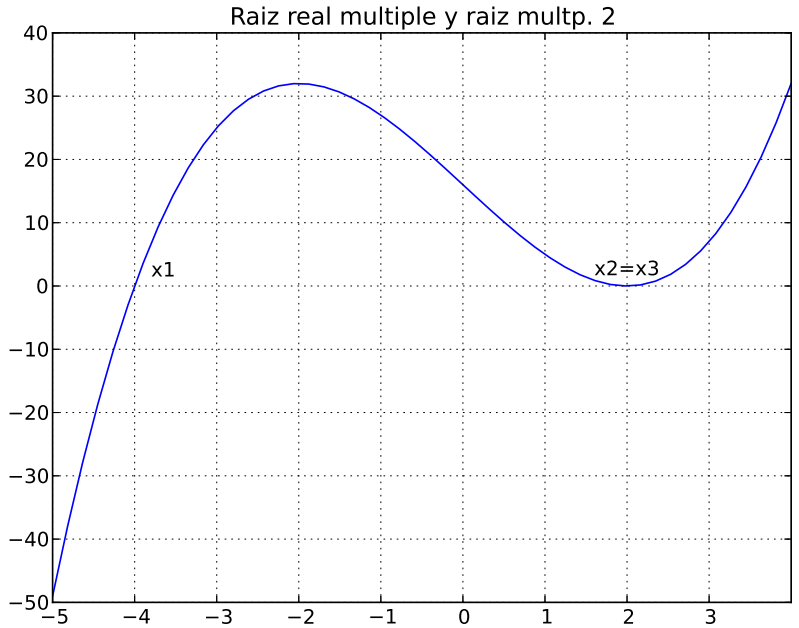


Raíz real simple y una raíz real con multiplicidad 2

$$\begin{aligned} f(x) &= x^3 - 12x + 16 \\ &= (x+4)(x-2)^2 \end{aligned}$$

Las raíces son:

$$\begin{aligned} x_1 &= -4 \\ x_2 &= 2 \\ x_3 &= 2 \end{aligned}$$

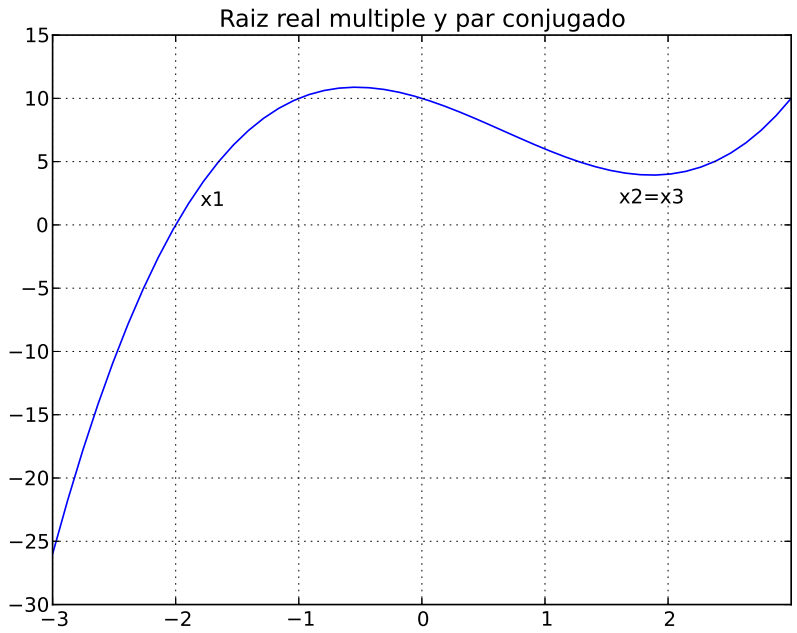


Raíz real y un par conjugado complejo

$$\begin{aligned} f(x) =& x^3 - 2x^2 - 3x + 10 \\ =& (x+2)(x-(2+i)) * \\ & *(x-(2-i)) \end{aligned}$$

Las raíces son:

$$\begin{aligned} x_1 &= -2 \\ x_2 &= 2+i \\ x_3 &= 2-i \end{aligned}$$



Contenido

Funciones algebraicas

Sea $g = f(x)$ la función expresada como

$$f_n y^n + f_{n-1} y^{n-1} + \ldots + f_1 y + f_0 = 0$$

Donde f_i es un polinomio de orden i en x.

Los polinomios son un caso simple de funciones algebraicas que se representan generalmente como

$$f_n(x) = a_0 + a_1 x + a_2 x^2 + \ldots + a_n x^n$$

Donde n es el orden del polinomio.

Contenido

Funciones trascedentales

Son aquellas que no son algebraicas.

Comprenden a las funciones trigonométricas, exponenciales, logarítmicas, entre otras.

Ejemplos:

- $$f(x) = ln(x^2 - 1)$$

- $$g(x) = e^{-0.2x} \sin(3x - 5)$$

Los métodos numéricos estándar para encontrar raíces pueden clasificarse en dos rubros:

1. La determinación de las raíces reales de ecuaciones algebraicas y trascendentales. Las técnicas a emplear en estos casos se diseñaron con el fin de encontrar el valor de una raíz simple de acuerdo con un conocimiento previo de su posición aproximada.

2. La determinación de todas las raíces reales y complejas de un polinomio, para lo cual los métodos numéricos estén diseñados específicamente para polinomios.

Determinan sistemáticamente todas las raíces del polinomio en lugar de hacerlo sólo con una, dada la posición aproximada.

Contenido

Método de incrementos sucesivos

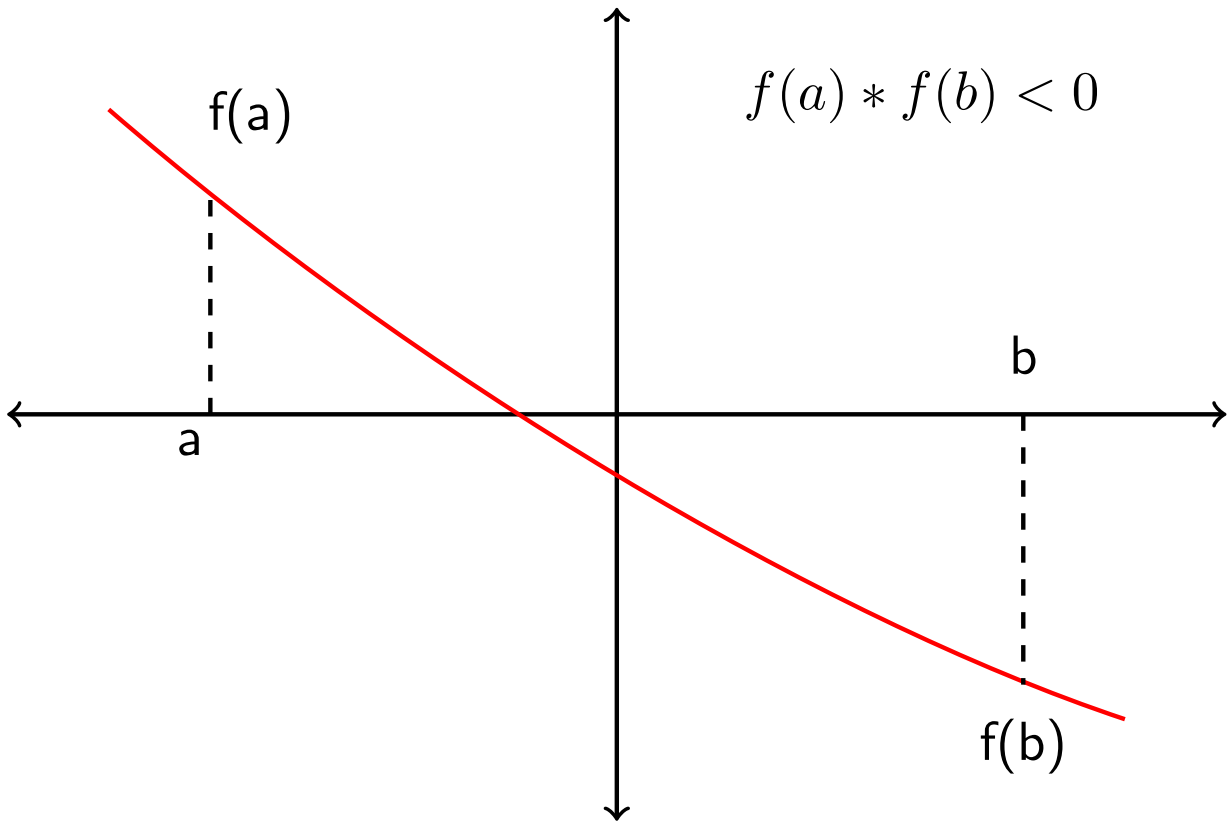
Podemos aproximar mucho mejor las raíces de una función, cuando la graficamos.

Con una gráfica general de unos cuantos puntos, tendríamos lo necesario para considerar los valores de las raíces.

El método de búsqueda incremental es una herramienta útil que podemos adoptar en conjunto con otras estrategias de cálculo de raíces, por sí sólo, éste método no nos ofrece más que una referencia sobre en dónde podrían estar esas raíces.

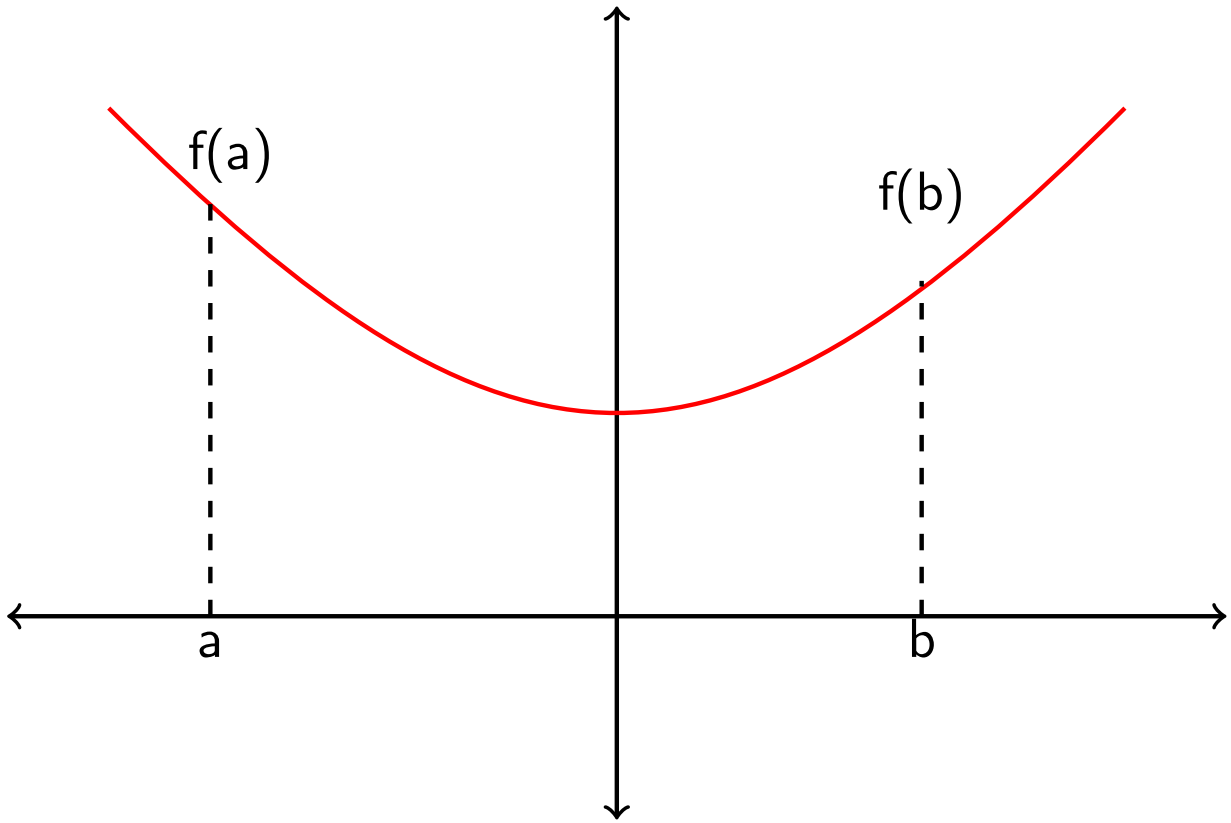
La idea básica detrás del método de búsqueda incremental es simple: si $f(x_1)$ y $f(x_2)$ tienen signos opuestos, entonces hay al menos una raíz en el intervalo (x_1, x_2).

Caso en donde es posible encontrar la raíz



Caso en donde no es posible encontrar la raíz

$$f(a) * f(b) > 0$$

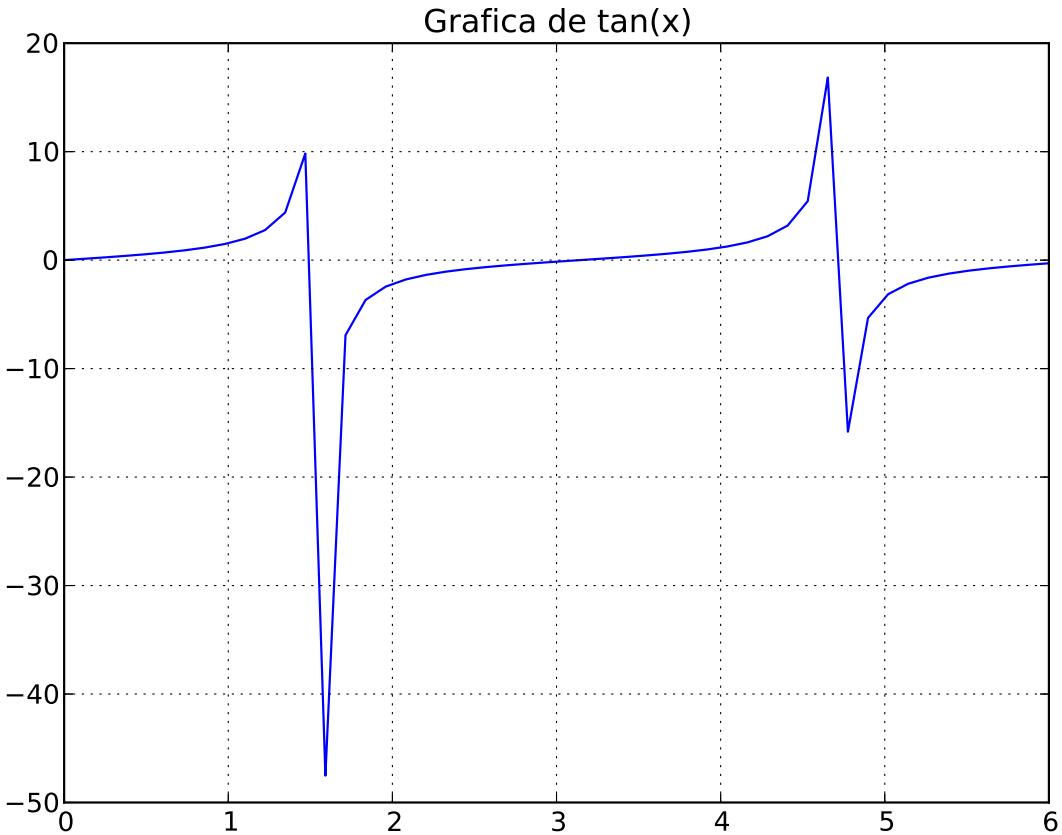


Si el intervalo es lo suficientemente pequeño, es probable que contenga una sola raíz. Así, los ceros de $f(x)$ puede ser detectados mediante la evaluación de la función a intervalos Δx y mirando cuando se presente un cambio de signo en la función.

Hay varios problemas con el método de búsqueda incremental:

1. Es posible perder dos raíces muy próximas entre sí, si el incremento de búsqueda Δx es mayor que la separación de las raíces.
2. Una raíz doble (dos raíces que coinciden) no será detectada.
3. Algunas singularidades de $f(x)$ se puede confundir con raíces. Por ejemplo, $f(x) = \tan x$. Tiene cambios de signo en $x = \pm 1/2n\pi$ con $n = 1, 3, 5, \ldots$

Estos puntos no son ceros verdaderos, ya que la función no cruza el eje x.



Código Método de incrementos sucesivos

El código busca un cero de la función f que proporciona el usuario en el intervalo (a, b) en incrementos de dx.

Se devuelve el intervalo (x_1, x_2) donde se encuentra la raíz, si la búsqueda se ha realizado correctamente; se devuelve $x_1 = x_2 =$ None cuando no se encontraron raíces.

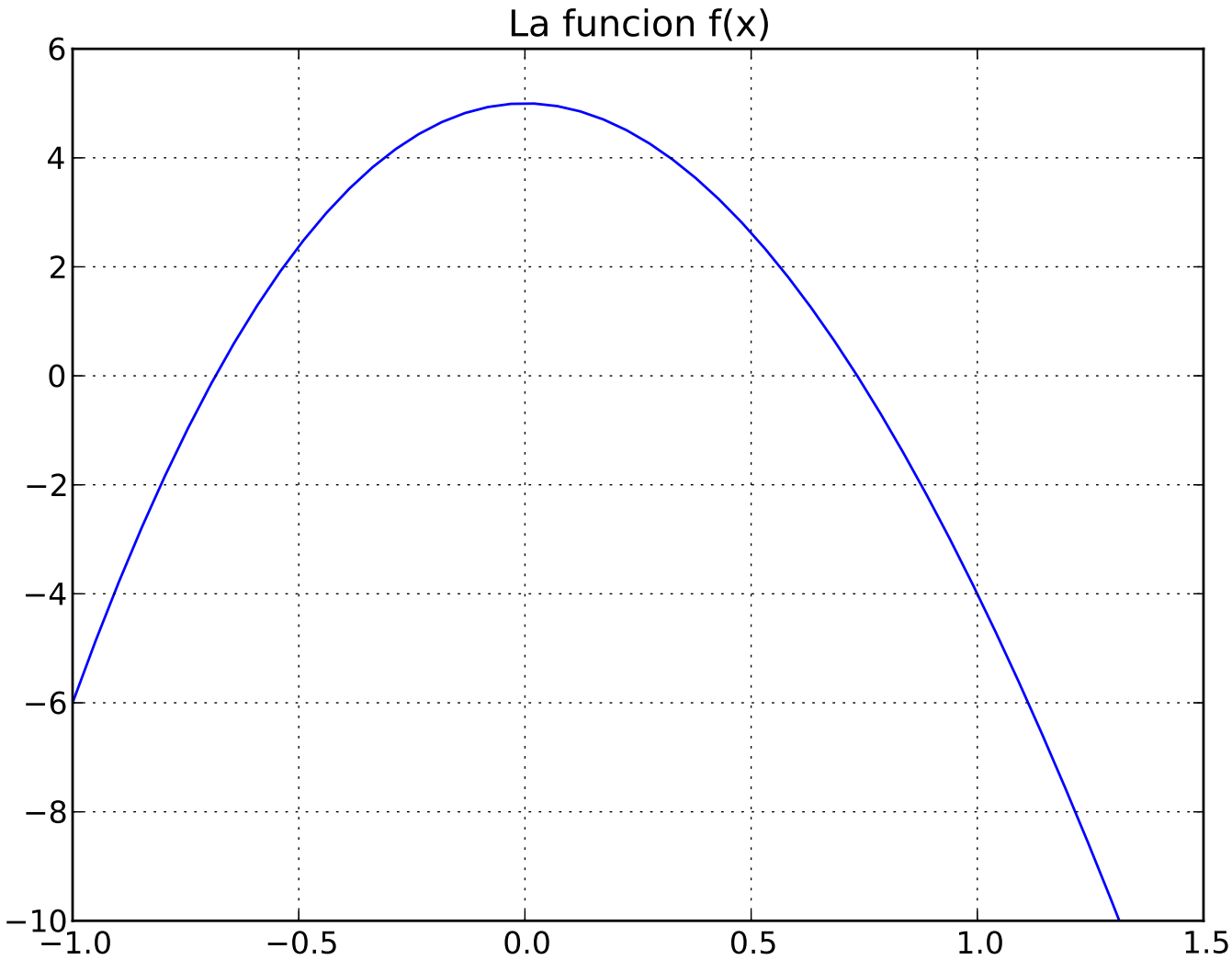
Luego de que se encontró la primera raíz, (la más cercana al punto a), se puede llamar de nuevo al procedimiento, sustitiyendo x_2 con el fin de encontrar la siguiente raíz. Esto se puede repetir siempre y cuando se detecta una raíz.

```
1 def buscaraiz(f,a,b,dx):
2     x1 = a; f1 = f(a)
3     x2 = a + dx; f2 = f(x2)
4     while f1*f2 > 0.0:
5         if x1 >= b: return None
6         x1 = x2; f1 = f2
7         x2 = x1 + dx; f2 = f(x2)
8     else:
9         return x1,x2
```

Ejemplo

Usa el método de incrementos sucesivos y con $\Delta x = 0.2$, para estimar la raíz con el valor positivo más pequeño de la función:

$$f(x) = x^3 - 10x^2 + 5$$


La funcion f(x)
6
4
2
0
−2
−4
−6
−8
−10
−1.0
−0.5
0.0
0.5
1.0
1.5

```
1 def f(x): return x**3 - 10*x**2 + 5.
2
3 a, b, dx = (0.0,1.5, 0.2)
4
5 print 'El intervalo es: '
6 x1, x2 = buscaraiz(f,a,b,dx)
7 print x1,x2
```

Contenido

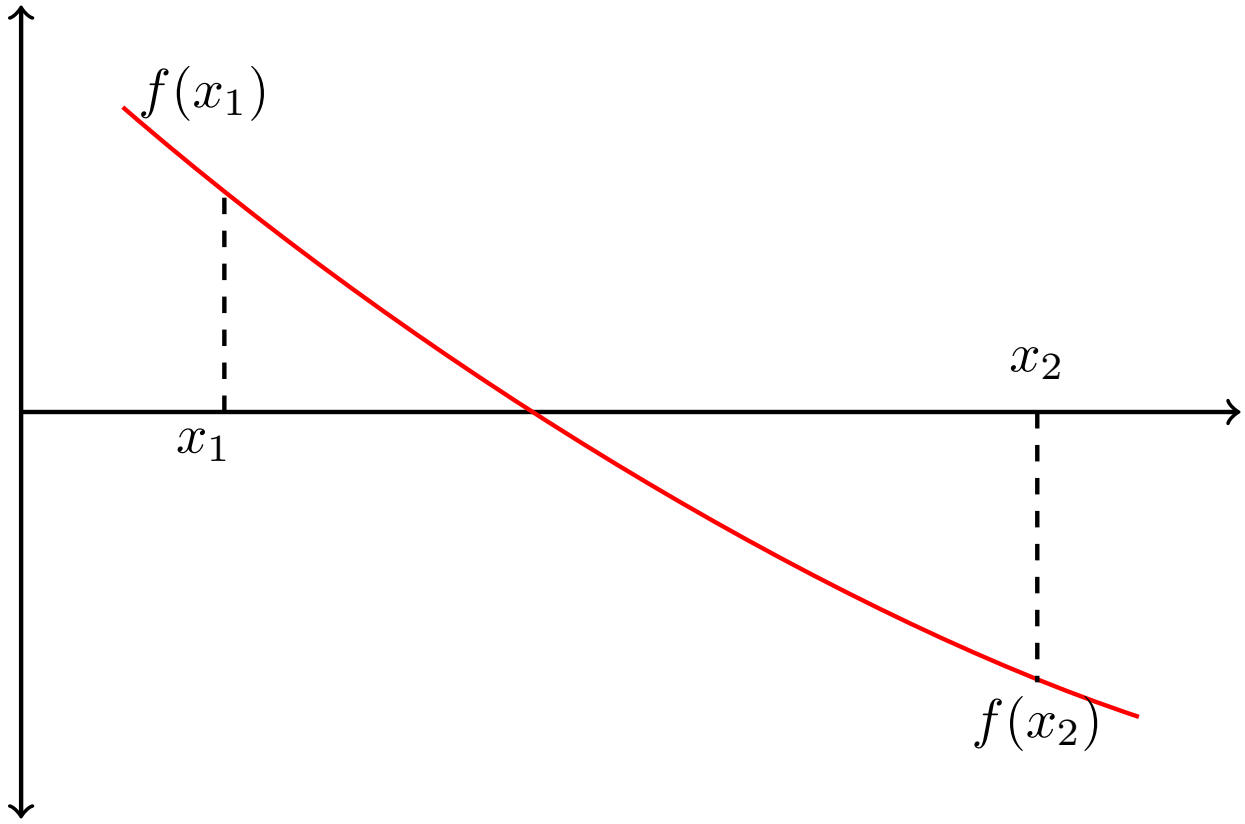
Método de Bisección

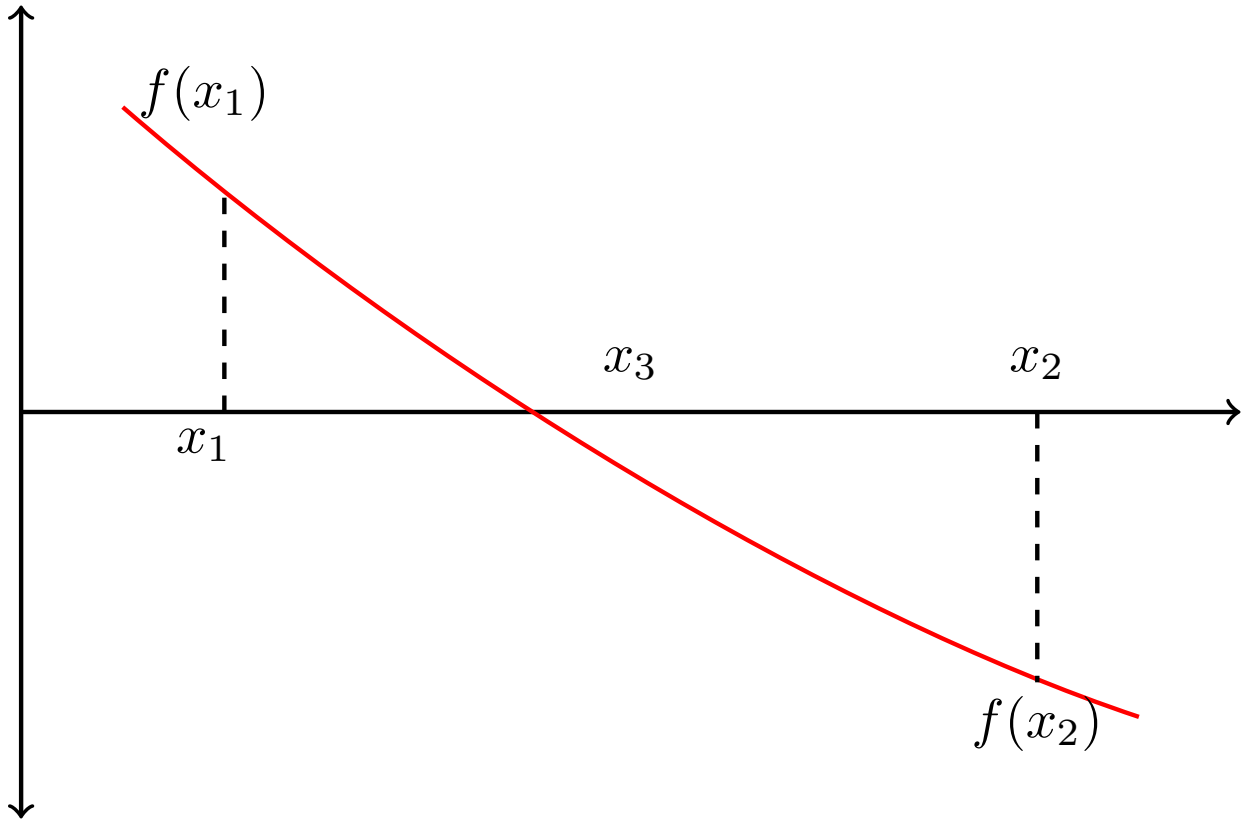
Después de que se ha identificado una raíz $f(x) = 0$ en el intervalo (x_1, x_2), disponemos de varios métodos para encontrar el valor de la raíz.

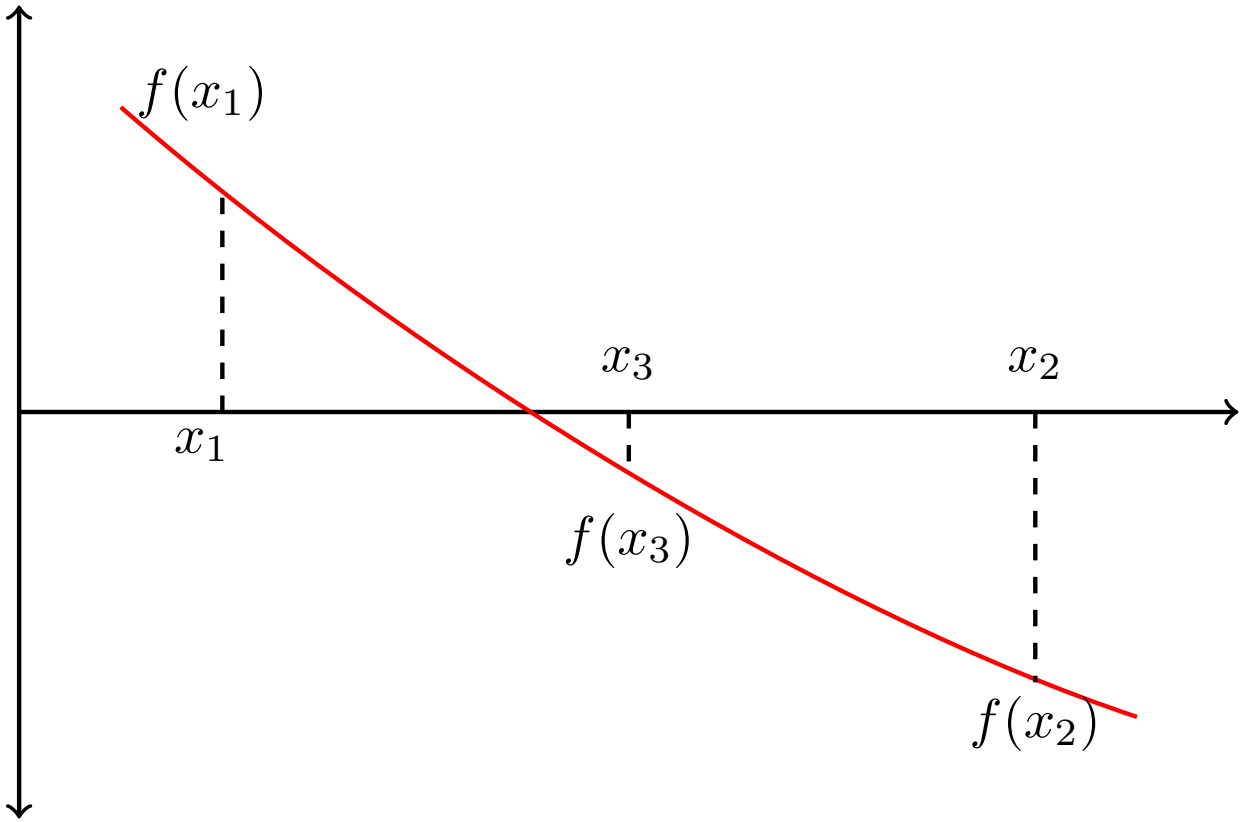
El método de bisección logra esta tarea: el intervalo se reduce sucesivamente a la mitad hasta que se vuelve suficientemente pequeño. La técnica de bisección no es el método más rápido disponible, pero es el más fiable. Una vez que una raíz se ha encontrado en un intervalo, nos podemos acercar a ella.

El método de bisección utiliza el mismo principio que el incremento sucesivo: si hay una raíz en el intervalo (x_1, x_2), entonces $f(x_1) * f(x_2) < 0$.

Con el fin de reducir a la mitad el intervalo, se calcula $f(x_3)$, donde $x_3 = (x1 + x2)/2$ es el punto medio del intervalo.

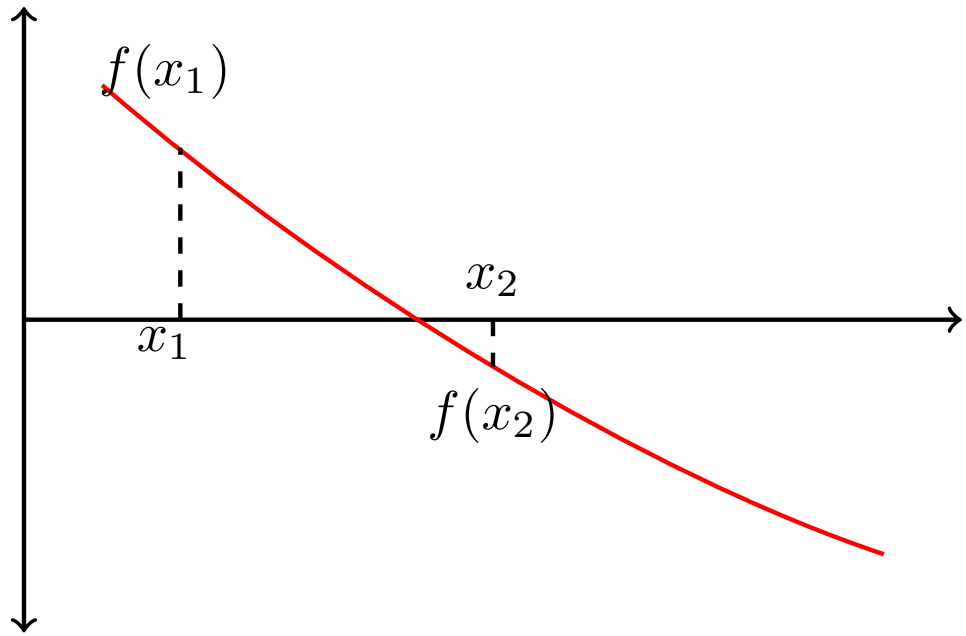

$f(x_1)$
x_2
x_1
$f(x_2)$


$f(x_1)$
x_3
x_2
x_1
$f(x_2)$

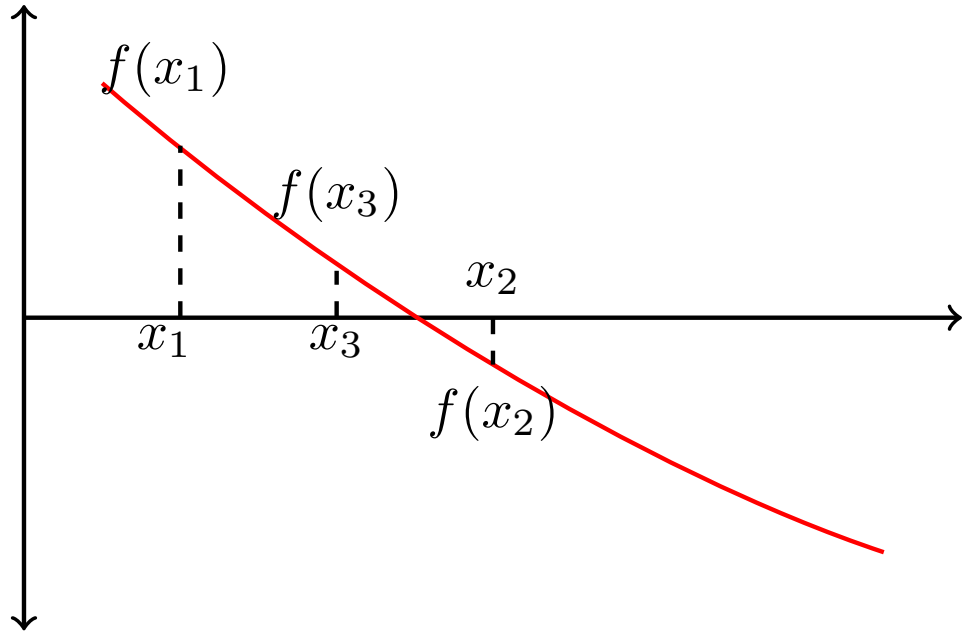

$f(x_1)$
x_3
x_2
x_1
$f(x_3)$
$f(x_2)$

Si $f(x_1) * f(x_3) < 0$, entonces la raíz debe estar en (x_1, x_3) entonces re-emplazamos del intervalo inicial x_2 por x_3. De lo contrario, la raíz se encuentra en (x_3, x_2), en tal caso, se sustituye x_3 por x_1.

Si $f(x_1) * f(x_3) < 0$, entonces la raíz debe estar en (x_1, x_3) entonces re-emplazamos del intervalo inicial x_2 por x_3. De lo contrario, la raíz se encuentra en (x_3, x_2), en tal caso, se sustituye x_3 por x_1.



Si $f(x_1) * f(x_3) < 0$, entonces la raíz debe estar en (x_1, x_3) entonces re-emplazamos del intervalo inicial x_2 por x_3. De lo contrario, la raíz se encuentra en (x_3, x_2), en tal caso, se sustituye x_3 por x_1.



En cualquiera de los casos, el nuevo intervalo (x_1, x_2) es la mitad del tamaño del intervalo original. La bisección es repite hasta que el intervalo se ha reducido a un valor ϵ pequeño, de modo que

$$|x_2 - x_1| \leq \epsilon$$

Es fácil calcular el número de biseccciones necesarias para alcanzar el valor de ϵ. El intervalo inicial Δx, se reduce a $\Delta x/2$ en la primera bisección, $\Delta x/2^2$ en la segunda, luego de n biseccciones, $\Delta x/2^n$.
Haciendo $\Delta x/2^n = \epsilon$, resolvemos para n

$$n = \frac{ln(|\Delta x|/\epsilon)}{ln2}$$

Algoritmo para el método de bisección

```
1 def bisect(f,x1,x2,switch,epsilon=1.0e-9):
2     f1 = f(x1)
3     if f1 == 0.0: return x1
4     f2 = f(x2)
5     if f2 == 0.0: return x2
6
7     if f1*f2 > 0.0: print 'La raiz no se ha
          identificado en un intervalo'
8
9     n = ceil(log(abs(x2 - x1)/epsilon)/log
          (2.0))
```

La función `ceil` devuelve el menor entero mayor o igual a x.

Ejemplos:

```
>>>ceil(1.1) # 2.0
>>>ceil(1.6) # 2.0
>>>ceil(-1.1) # -1.0
>>>ceil(-1.6) # -1.0
```

Algoritmo para el método de bisección

```
1       for i in np.arange(n):
2           x3 = 0.5*(x1 + x2); f3 = f(x3)
3           if (switch == 0) and (abs(f3) > abs(f1
                ))) \
4                                   and (abs(f3) > abs(f2)
                                        ):
5               return None
6           if f3 == 0.0: return x3
7           if f2*f3 < 0.0:
8               x1 = x3; f1 = f3
9           else:
10              x2 =x3; f2 = f3
11      return (x1 + x2)/2.0
```

Haciendo que la variable `switch` $= 1$, se forza la rutina para comprobar si la magnitud de $f(x)$ disminuye con la reducción a la mitad de cada intervalo.

Si no es así, algo anda mal (probablemente la "raíz" no es una raíz en absoluto, sino una singularidad) y se devuelve la `raíz` $=$ `None`. Dado que esta característica no siempre es deseable, el valor predeterminado es `switch`=0.

Veamos lo que hace la variable `switch`:

Hasta el momento no nos hemos fijado en la magnitud de $f(x_1)$ y $f(x_2)$, sólo y en el signo del respectivo producto, pero ahora, tomamos en cuenta la magnitud tanto de los puntos evaluados en la función como del producto mismo.

Consideremos le primer caso de la función $f(x)$, donde ya sabemos que hay una raíz en el intervalo $[0.6, 0.8]$

El valor por defecto de `switch` = 0

x_1	x_2	$f(x_1)$	$f(x_2)$	$f(x_1)f(x_2)$
0.0	0.2	5.0	4.608	23.04
0.2	0.4	4.608	3.464	15.9621
0.4	0.6	3.464	1.616	5.5978
0.6	0.8	1.616	-0.888	-1.4350
0.8	1.0	-0.888	-4.0	3.552
1.0	1.2	-4.0	-7.672	30.688

Del código tenemos que:

```
if (switch == 0) and (abs(f3) > abs(f1)) \
                 and (abs(f3) > abs(f2)):
    return None
```

$abs(f3) > abs(f1)$	$abs(f3) > abs(f2)$	Expresión
True	True	True

Aquí tendrán que completar la tabla de tal manera que revisen el resultado de la expresión compuesta.

Ejercicios

Implementa el método de bisección para calcular el(los) intervalo(s) y la(s) raíz(ces) de:

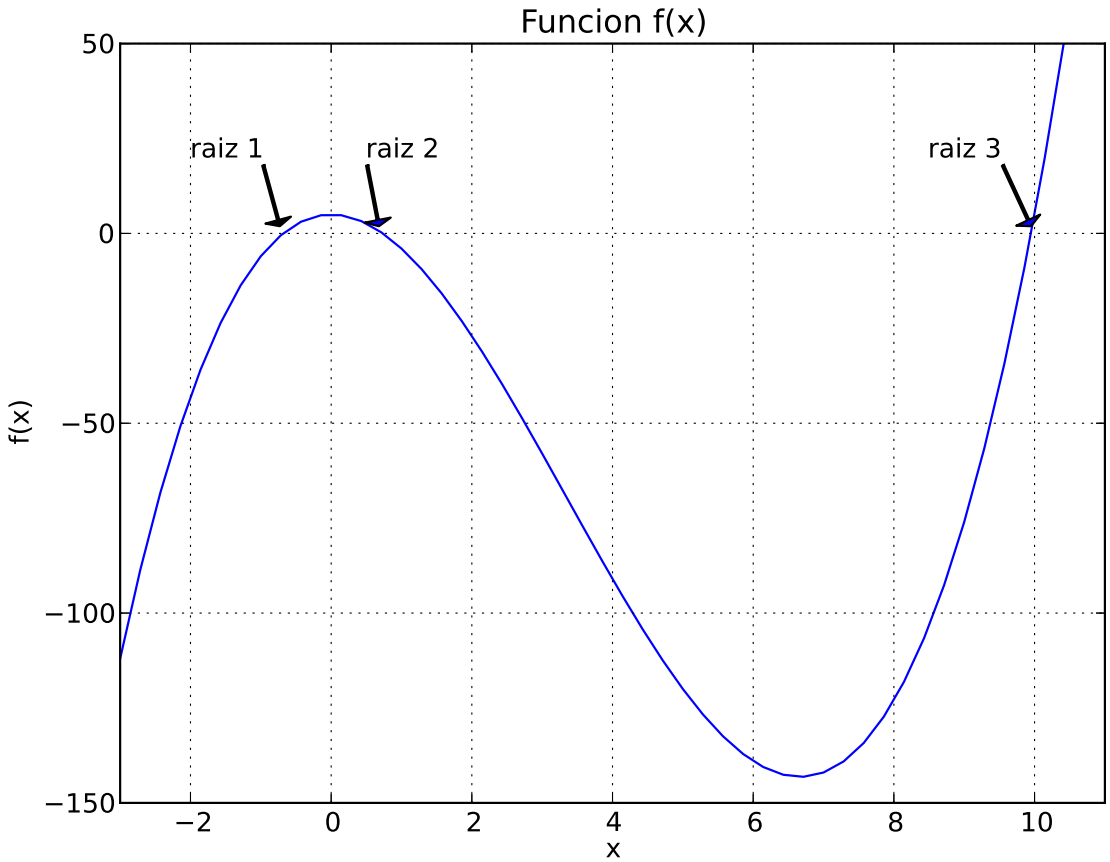
$$f(x) = x^3 - 10x^2 + 5$$
$$g(x) = x - \tan(x) \qquad 0 \leq x \leq 20$$

Con una tolerancia de 1×10^{-9}

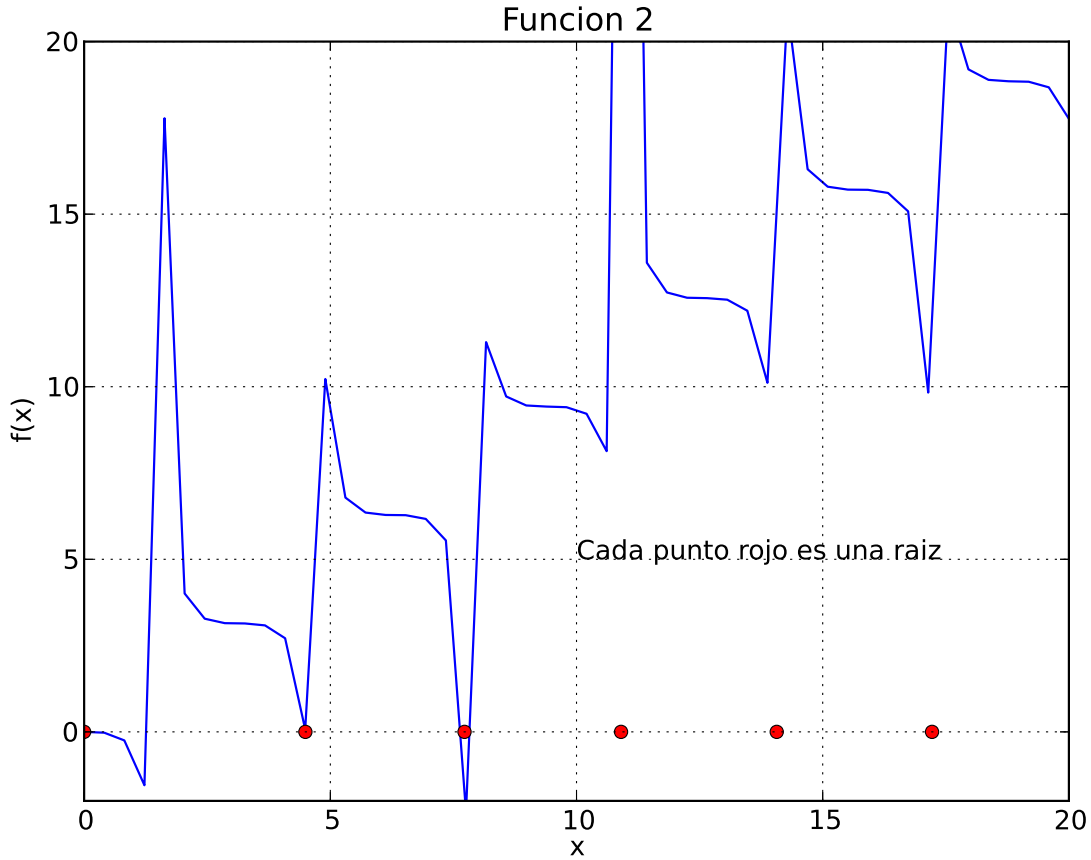
Toma en cuenta que $\tan(x)$ es singular y cambia de signo en $x = \pi/2, 3\pi/2, \ldots$. Para no confundir estos puntos para las raíces, hacemos `switch`$= 1$.

La proximidad de las raíces a las singularidades es otro problema potencial que puede ser prevenido mediante el uso de Δx pequeña, usa $x = 0.01$.

Graficas de las funciones y sus raíces



Graficas de las funciones y sus raíces



Estrategia de solución

Para integrar los elementos que hemos visto:

1. Recuperamos el algoritmo que nos identifica en qué intervalo se encuentra una raíz.
2. Aplicamos el algoritmo del método de bisección.
3. Usamos un ciclo que nos revise en el dominio que se nos proporciona, si existe más de una raíz.

```
1  def f(x): return x**3-10*x**2+5
2
3  a,b,dx = (-2.0,11.0,0.02)
4
5  print 'Intervalo (x1,x2)    raiz'
6  while 1:
7      try:
8          x1, x2 = buscaraiz(f,a,b,dx)
9      except Exception, e:
10         print e; break
11     if x1 != None:
12         a = x2
13         root = bisect(f,x1,x2,0)
14         if raiz != None: print '(%2.4f, %2.4f)
               %2.8f' %(x1, x2, raiz)
```

```
1       else:
2           print '\nHecho'
3           break
```

Contenido

Método de Newton-Raphson

El método de Newton-Raphson es el algoritmo más conocido para encontrar raíces por una buena razón: es simple y rápido.

El único detalle es que utiliza la derivada $f'(x)$ así como la función $f(x)$. Por tanto, en los problemas a resolver con este algoritmo, deberá de contemplarse que la derivada sea fácil de calcularse.

El método de N-R se obtiene de la expansión en series de Taylor de $f(x)$ alrededor de x:

$$f(x_{i+1}) = f(x_i) + f'(x_i)(x_{i+1} - x_i) + O(x_{i+1} - x_i)^2$$

Si x_{i+1} es una raíz de $f(x) = 0$, tenemos que:

$$0 = f(x_i) + f'(x_i)(x_{i+1} - x_i) + O(x_{i+1} - x_i)^2$$

Suponiendo que x_i está cerca de x_{i+1}, podemos eliminar el último término de la ecuación y resolver para x_{i+1}, por lo que la fórmula de Newton-Raphson es:

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

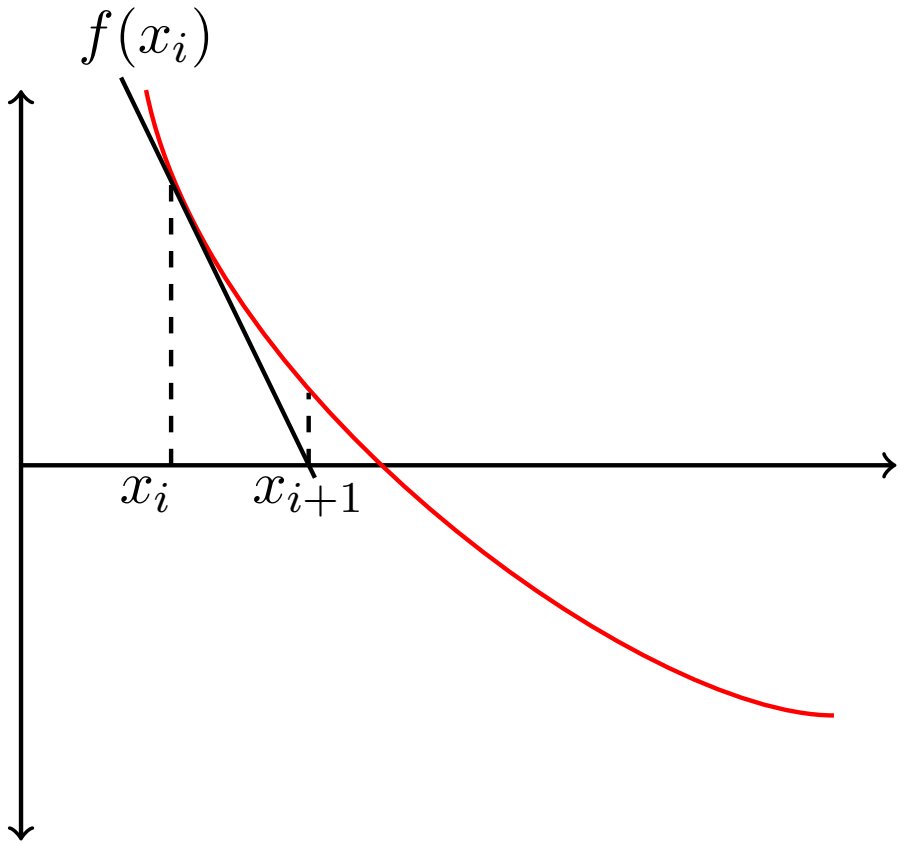
Si x representa el valor verdadero de la raíz, el error en x_i es $E_i = x - x_i$. Se puede demostrar que si x_{i+1} se calcula de la expresión de N-R, el error es:

$$E_{i+1} = -\frac{f''(x_i)}{2f'(x_i)}E_i^2$$

Lo que nos dice que el método de N-R converge de manera cuadrática, es decir, el error es el cuadrado del error del punto previo.

Representación gráfica

Podemos interpretar que x_{i+1} es el punto en donde la tangente de $f(x_i)$ cruza el eje de las x:



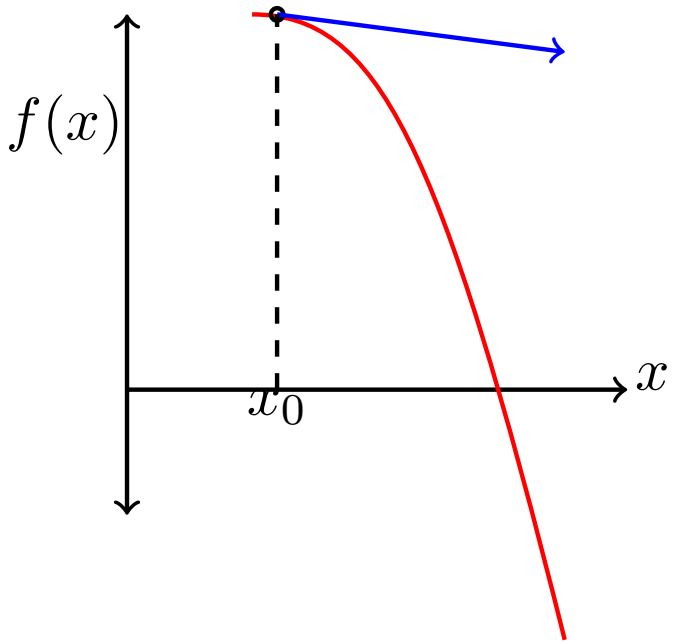
El método de N-R es sencillo: se aplica la expresión para x_{i+1} iniciando con un valor x_0, hasta alcanzar un criterio de convergencia:

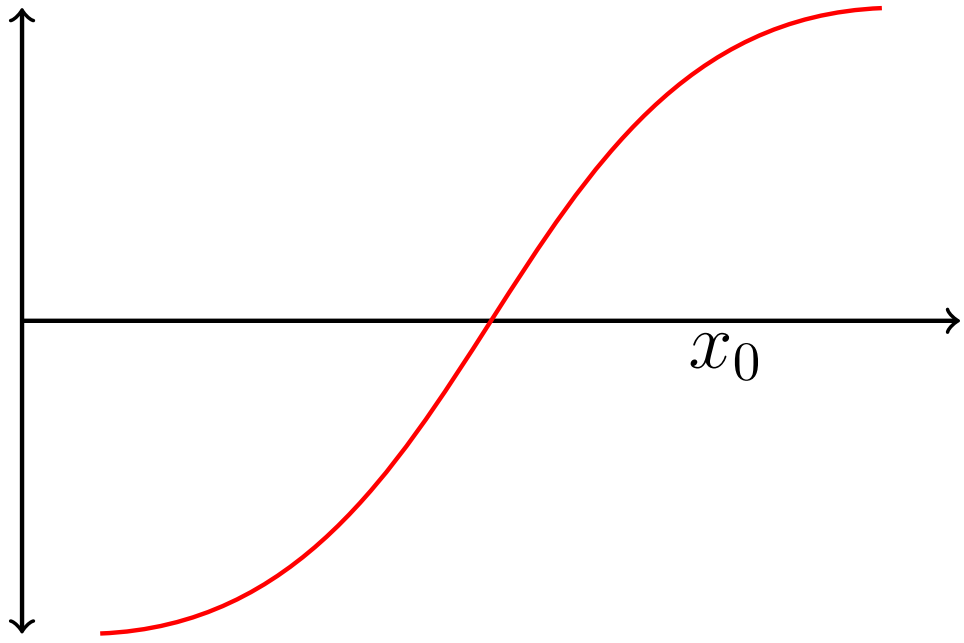
$$|x_{i+1} - x_i| < \epsilon$$

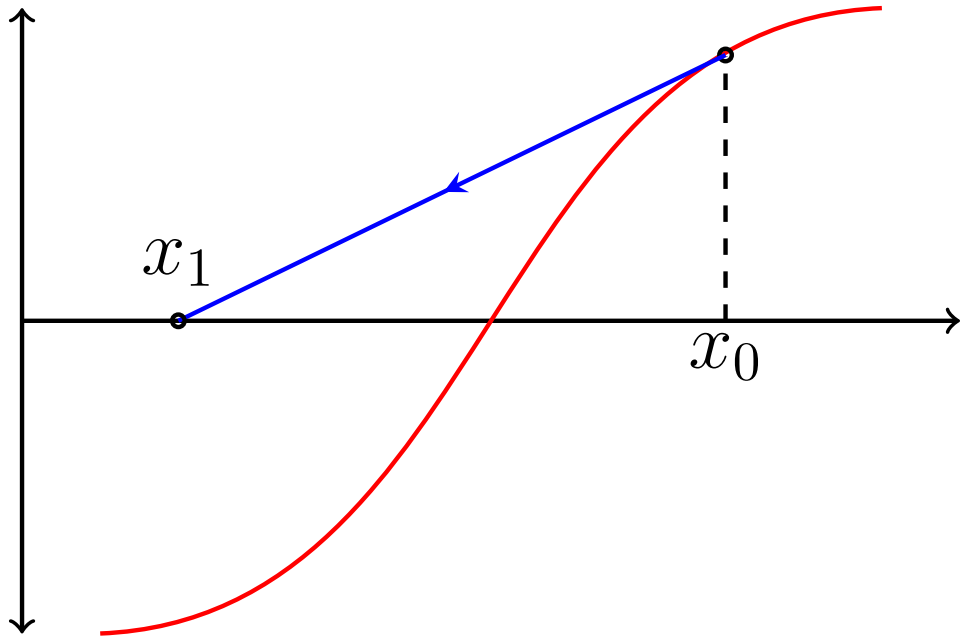
El algoritmo es el siguiente:

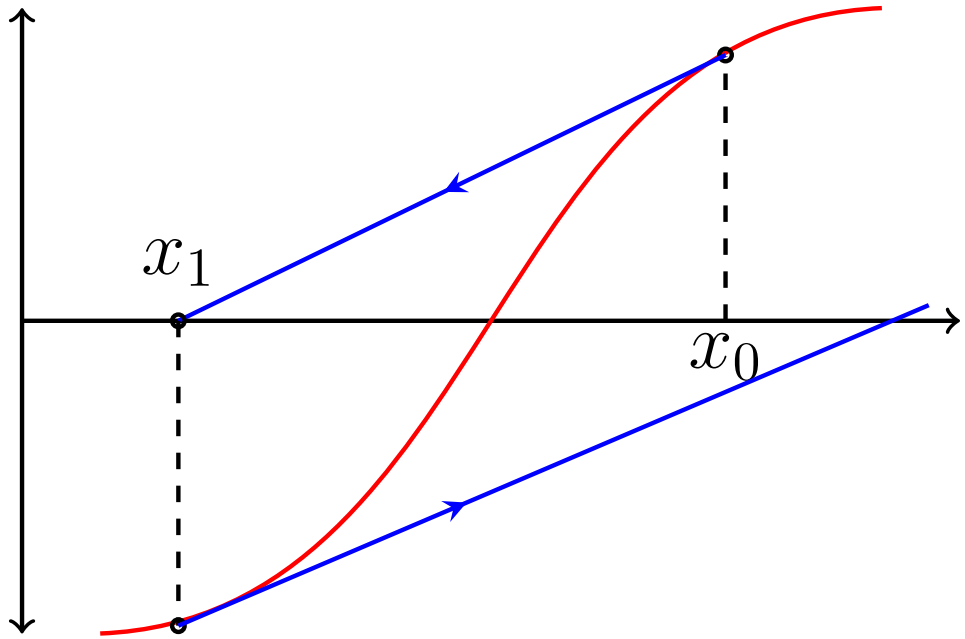
1. Sea x un valor inicial para la raíz de $f(x) = 0$.
2. Calcular $\Delta x = -f(x)/f'(x)$.
3. Asignar $x \leftarrow x + \Delta x$ y se repiten los pasos 2-3, hasta alcanzar $|\Delta x| < \epsilon$.

Aunque el método de Newton-Raphson converge rápidamente cerca de la raíz, sus características globales de convergencia son pobres. La razón es que la línea tangente no es siempre una aproximación aceptable de la función.


$f(x)$
x
x_0


x_0


x_1
x_0


x_1
x_0

Algoritmo para el método Newton-Raphson

La siguiente algoritmo para el método de Newton-Raphson supone que la raíz a calcularse inicialmente está en el intervalo (a, b).

El punto medio del intervalo se utiliza como aproximación inicial de la raíz. Los extremos del intervalo se actualizan luego de cada iteración. Si una iteración del método Newton-Raphson no se mantiene dentro del intervalo, se descarta y se reemplaza con el método de bisección.

Ya que el método `newtonRaphson` utiliza la función $f(x)$, así como su derivada, (denotadas por f y df) deben ser proporcionadas por el usuario.

Algoritmo de Newton-Raphson

```
1 def newtonRaphson(f,df,a,b,tol=1.0e-9):
2     fa = (f(a)
3     if fa == 0.0: return a
4     fb = f(b)
5     if f(b) == 0.0: return b
6     if fa*fb > 0.0: print 'La raiz no esta en
          el intervalo'
7     x = 0.5 * (a + b)
```

```
1        for i in range(30):
2            fx = f(x)
3            if abs(fx) < tol: return x
4
5            if fa*fx < 0.0:
6                b = x
7            else:
8                a = x; fa = fx
```

```
1           dfx = df(x)
2
3           try: dx = -fx/dfx
4           except ZeroDivisionError: dx = b - a
5           x =  x + dx
6
7           if(b - x)*(x - a) < 0.0:
8               dx = 0.5*(b-a)
9               x = a + dx
10
11          if abs(dx) < tol*max(abs(b),1.0):
                return x
12
13      print 'Son demasiadas iteraciones'
```

Ejercicio

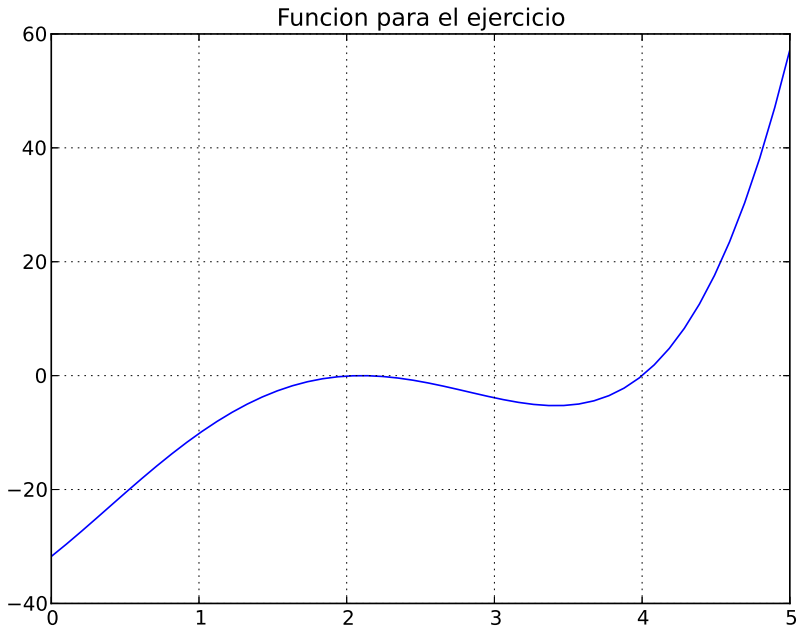
Encontrar la raíz positiva más pequeña de

$$f(x) = x^4 - 6.4x^3 + 6.45x^2 + 20.538x - 31.752$$

Ejercicio

Encontrar la raíz positiva más pequeña de

$$f(x) = x^4 - 6.4x^3 + 6.45x^2 + 20.538x - 31.752$$



```
1  def f(x): return x**4 - 6.4*x**3 + 6.45*x**2 +
       20.538*x - 31.752
2  def df(x): return 4.0*x**3 - 19.2*x**2 + 12.9*
     x + 20.538
3
4  def newtonRaphson(x,tol=1e-05):
5      for i in range(30):
6          dx = -f(x)/df(x)
7          x = x + dx
8          if abs(dx) < tol: return x,i
9      print 'Son demasiadas iteraciones\n'
10
11 raiz,numIter = newtonRaphson(2.0)
12
13 print 'Raiz =',raiz
14 print 'Numero de iteraciones =',numIter
```

Ejercicios

1. Encuentra las raíces de $x \sin x + 3 \cos x - x = 0$ en el intervalo $(-6, 6)$
2. Calcula todas las raíces reales de $x^4 + 0.9x^3 - 2.3x^2 + 3.6x - 25.2 = 0$.
3. Calcula todas las raíces reales de $x^4 + 2x^3 - 7x^2 + 3 = 0$.
4. Calcula todas las raíces de $\sin x - 0.1x = 0$.

Contenido

Método de la falsa posición

Este método es parecido al de bisección, ya que el intervalo que contiene a la raíz se va reduciendo.

En vez de bisectar de manera monótona el intervalo, se utiliza una interpolación lineal ajustada a dos puntos extremos para encontrar la aproximación de la raíz.

La función está bien aproximada por la interpolación lineal, con lo que las raíces tendrán una buena precisión; la iteración convergerá más rápido que como ocurre con el método de bisección.

Dado un intervalo $[a, c]$ que contenga a la raíz, la función lineal que pasa por $(a, f(a))$ y $(c, f(c))$ se escribe como:

$$y = f(a) + \frac{f(c) - f(a)}{c - a}(x - a)$$

de donde se despeja x:

$$x = a + \frac{c - a}{f(c) - f(a)}(y - f(a))$$

La coordenada x en donde la línea intersecta al eje x se determina al hacer $y = 0$ en la ecuación anterior, por tanto:

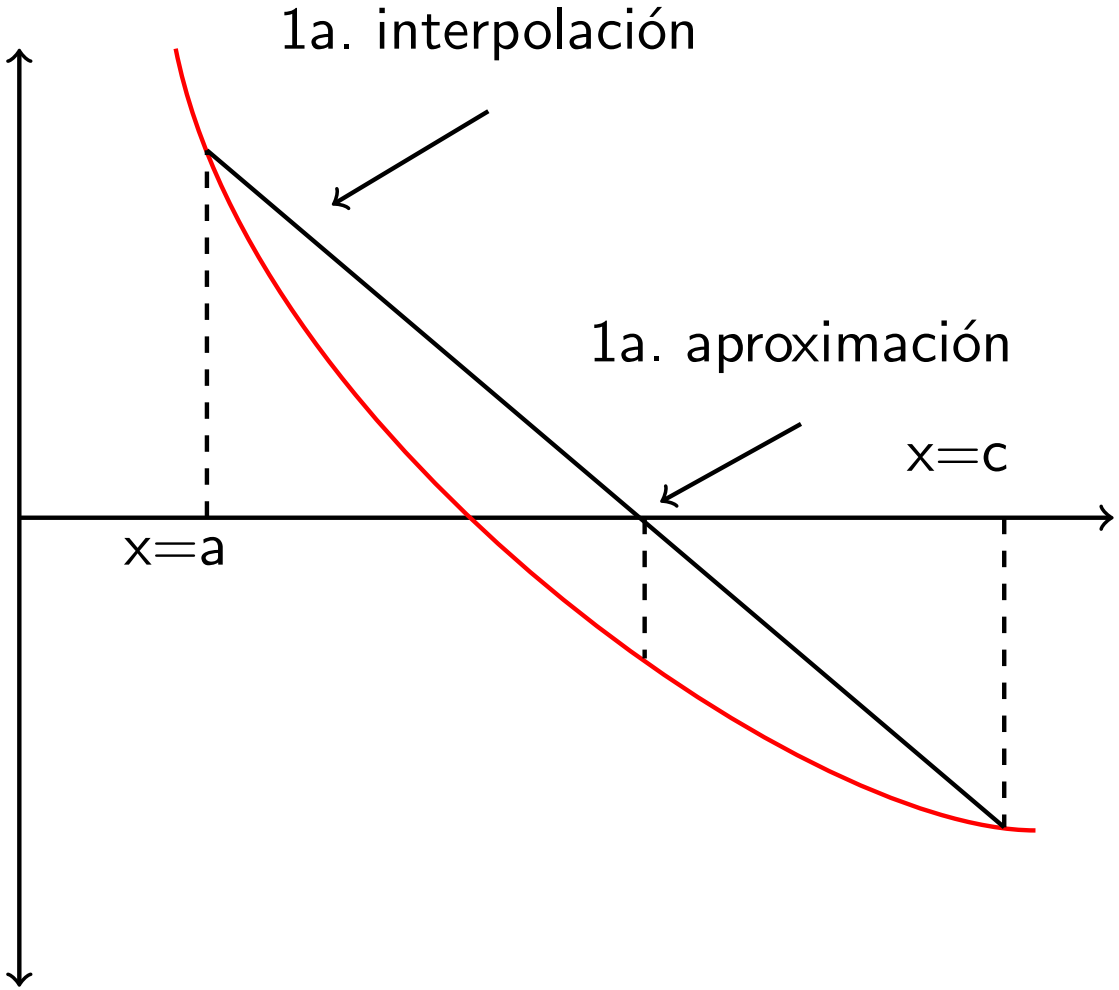
$$b = a - \frac{c - a}{f(c) - f(a)} f(a) = \frac{af(c) - cf(a)}{f(c) - f(a)}$$

Después de encontrar b, el intervalo $[a, c]$ se divide en $[a, b]$ y $[b, c]$.

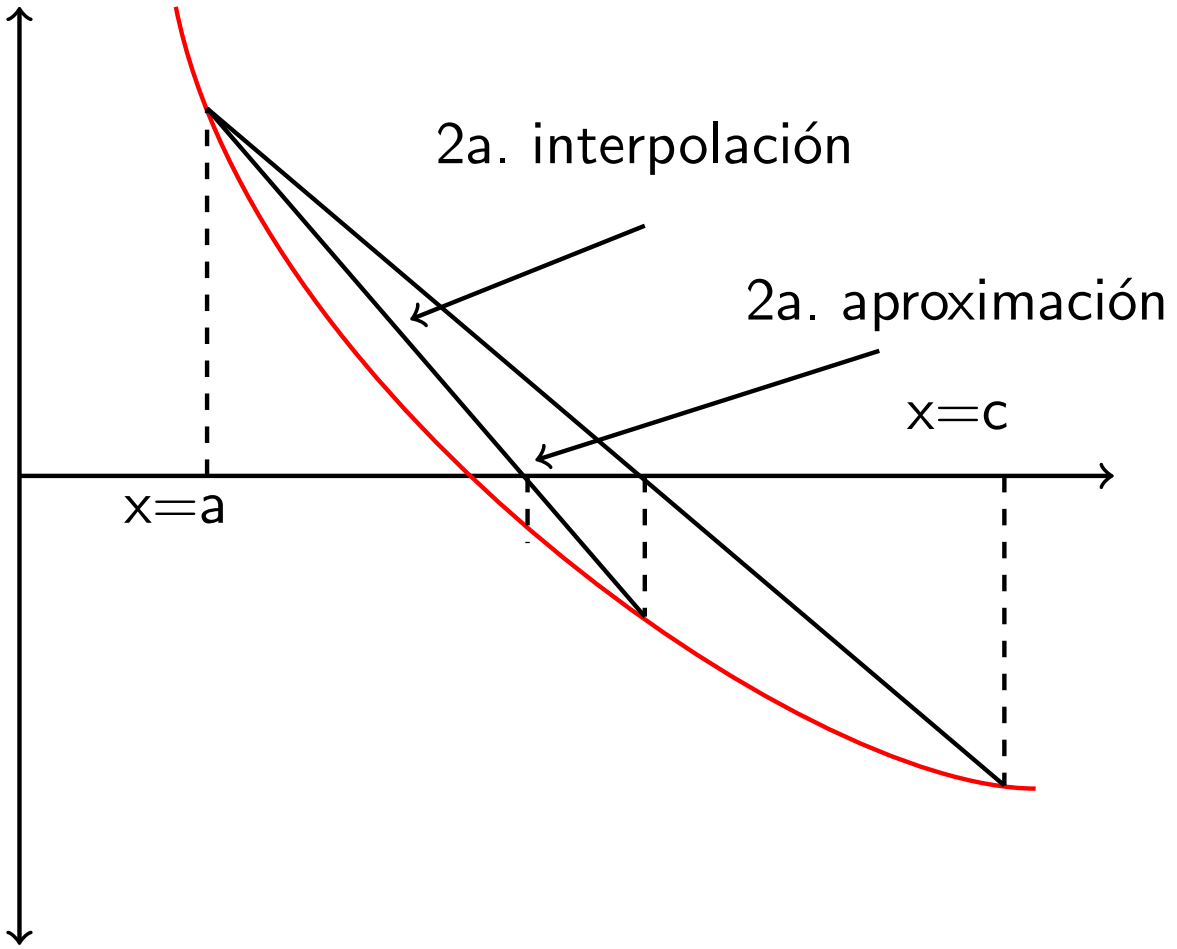
Si $f(a)f(b) \leq 0$, la raíz se encuentra en $[a, b]$; en caso contrario, está en $[b, c]$. Los extremos del nuevo intervalo que contiene a la raíz se renombran para el siguiente paso como a y c.

El procedimiento de interpolación se repite hasta que las raíces estimadas convergen.

Método de la falsa posición



Método de la falsa posición



La desventaja de este método es que aparecen extremos fijos: uno de los extremos de la sucesión de intervalos no se mueve del punto original, por lo que las aproximaciones a la raíz, denotadas por b_1, b_2, b_3, etc. convergen a la raíz exacta solamente por un lado.

Los extremos fijos no son deseables debido a que hacen más lenta la convergencia, en particular cuando el intervalo es grande o cuando la función se desvía de manera significativa de una línea recta en el intervalo.

¿Qué podemos hacer al respecto?

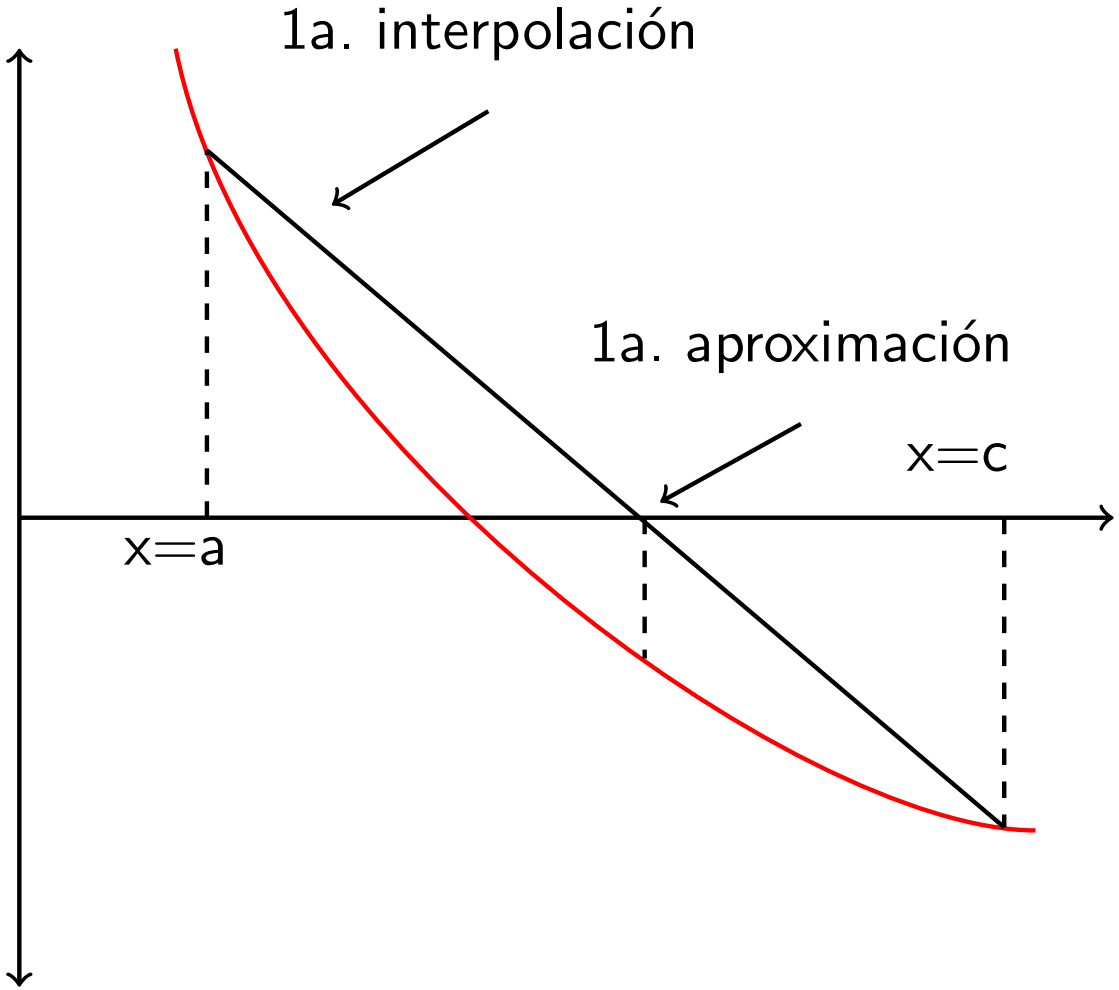
Contenido

Método de la falsa posición modificado

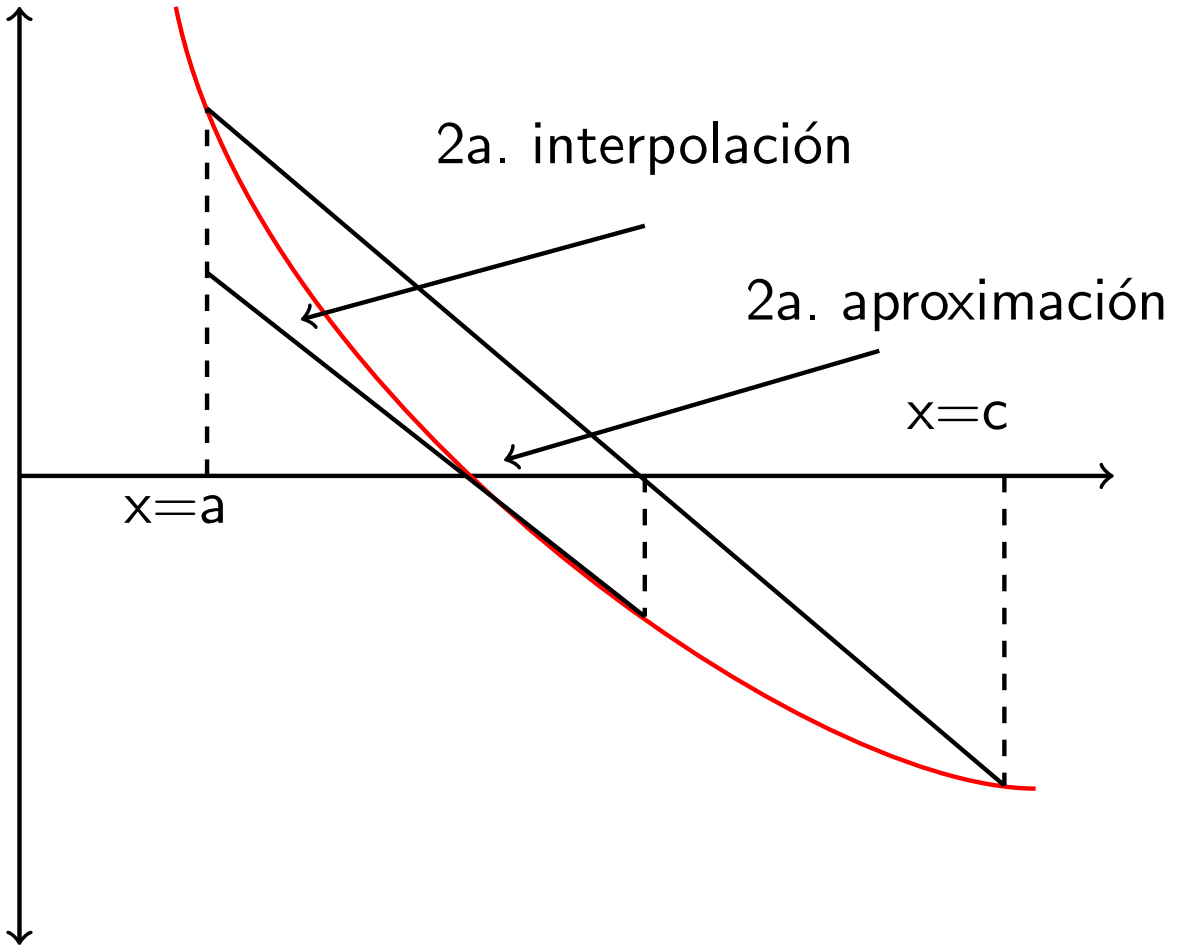
En este método, el valor de f en un punto fijo se divide a la mitad si este punto se ha repetido más de dos veces.

El extremo que se repite se llama extremo fijo. La excepción para esta regla es que para $i = 2$, el valor de f en un extremo se divide entre 2 de inmediato si no se mueve.

Método de falsa posición modificado



Método de la falsa posición modificado



Contenido

Método de la secante

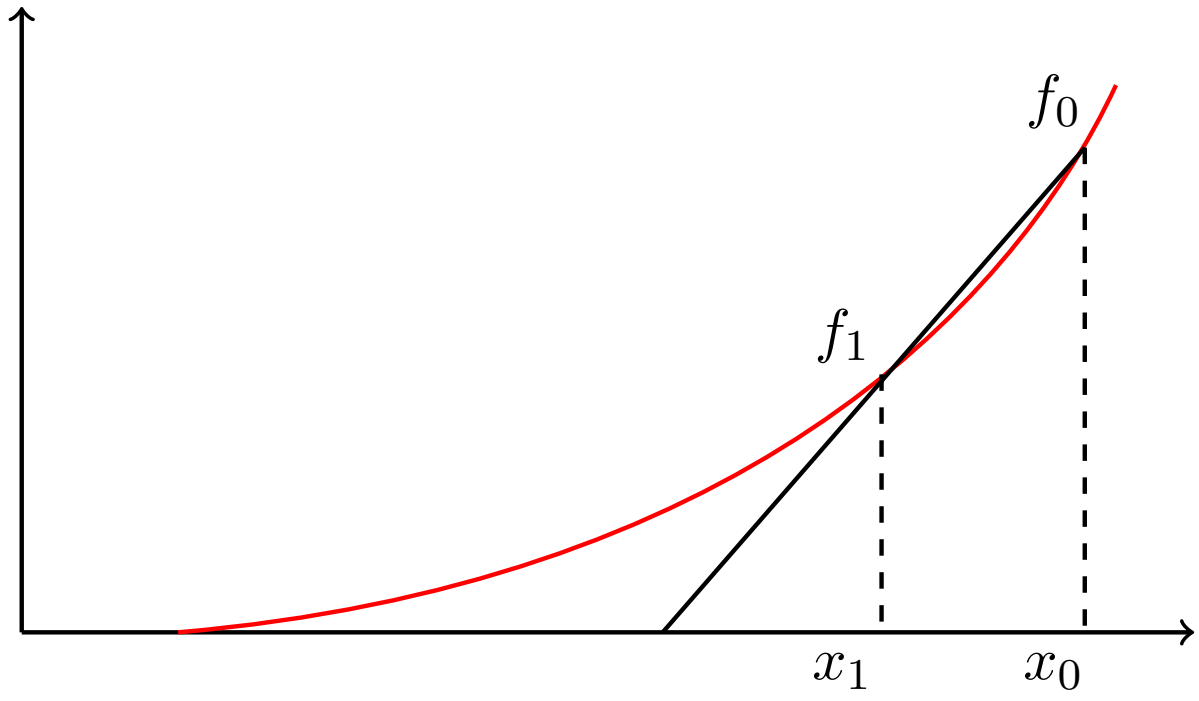
A diferencia del método de Newton, el valor de f' se aproxima utilizando dos valores de iteraciones consecutivas de f. Con lo que se elimina la necesidad de evaluar tanto a f como a f' en cada iteración.

Las aproximaciones sucesivas para la raíz en el método de la secante están dadas por:

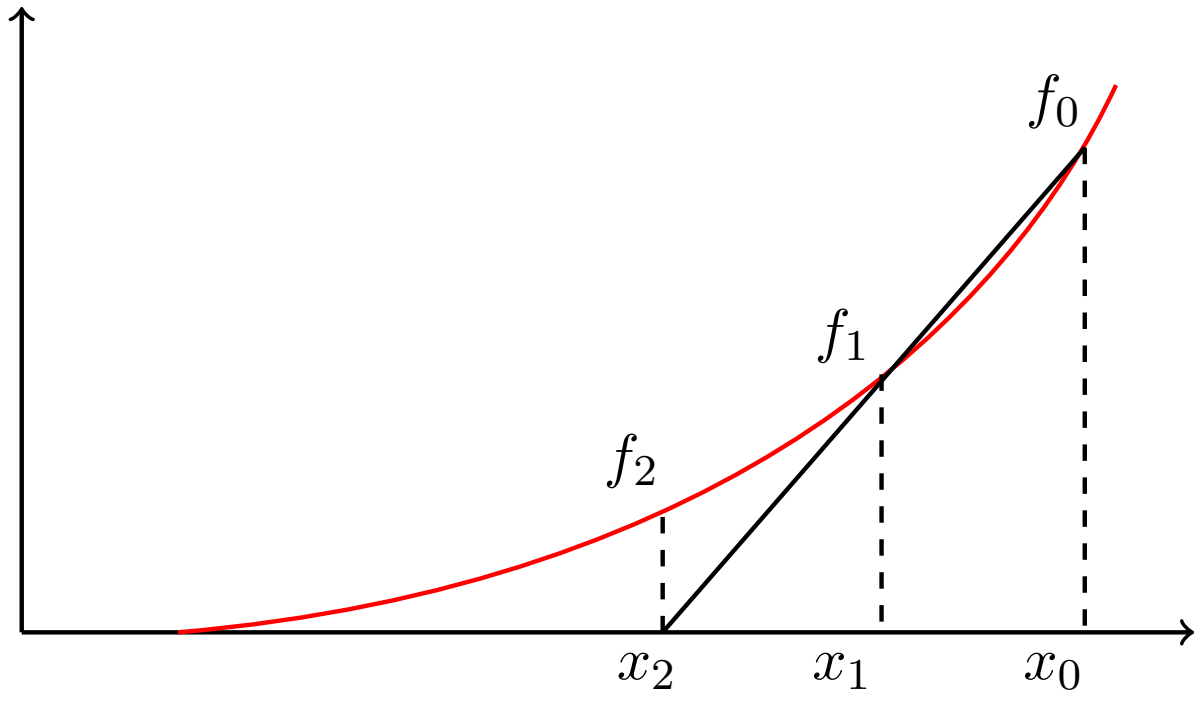
$$x_n = x_{n-1} - y_{n-1}\frac{x_{n-1} - x_{n-2}}{y_{n-1} - y_{n-2}}, \qquad n = 2, 3, \ldots$$

donde x_0 y x_1 son dos suposiciones iniciales para comenzar la iteración.

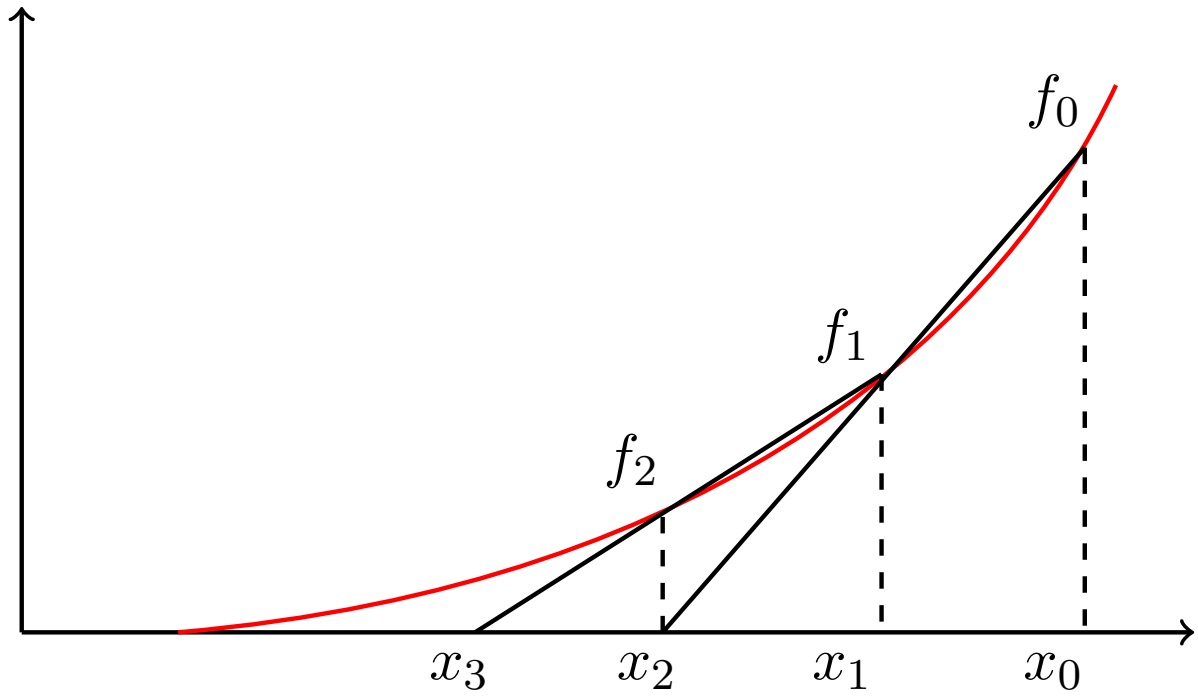
Método de la secante



Método de la secante



Método de la secante



Atento aviso

En las técnicas de falsa posición, falsa posición modificada y el método de la secante, no hemos presentado como tal un código en `Python` que nos devuelva las raíces, por lo que tendrás que proponer un código para cada una de las técnicas.

Ese código lo vas a utilizar par resolver los problmeas y ejercicios del examen.