Tema 2 - Operaciones matemáticas básicas

Cálculo de raíces

M. en C. Gustavo Contreras Mayén

Facultad de Ciencias - UNAM

25 de febrero de 2018

1. Bases generales

2. Funciones algebraicas y trascendentales

3. Método de incrementos sucesivos

4. Métodos para el cálculo de raíces

1. Bases generales
1.1 Cálculo de raíces
2. Funciones algebraicas y trascendentales
3. Método de incrementos sucesivos
4. Métodos para el cálculo de raíces

Cálculo de raíces

Sea $y = f(x)$.

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

Cálculo de raíces

El teorema fundamental del álgebra indica que todo polinomio de grado n, tiene n raíces.

En el caso de las raíces reales, corresponden a los valores de x que hacen que la función corte el eje de las abscisas:

Ejemplo de la función seno(x)

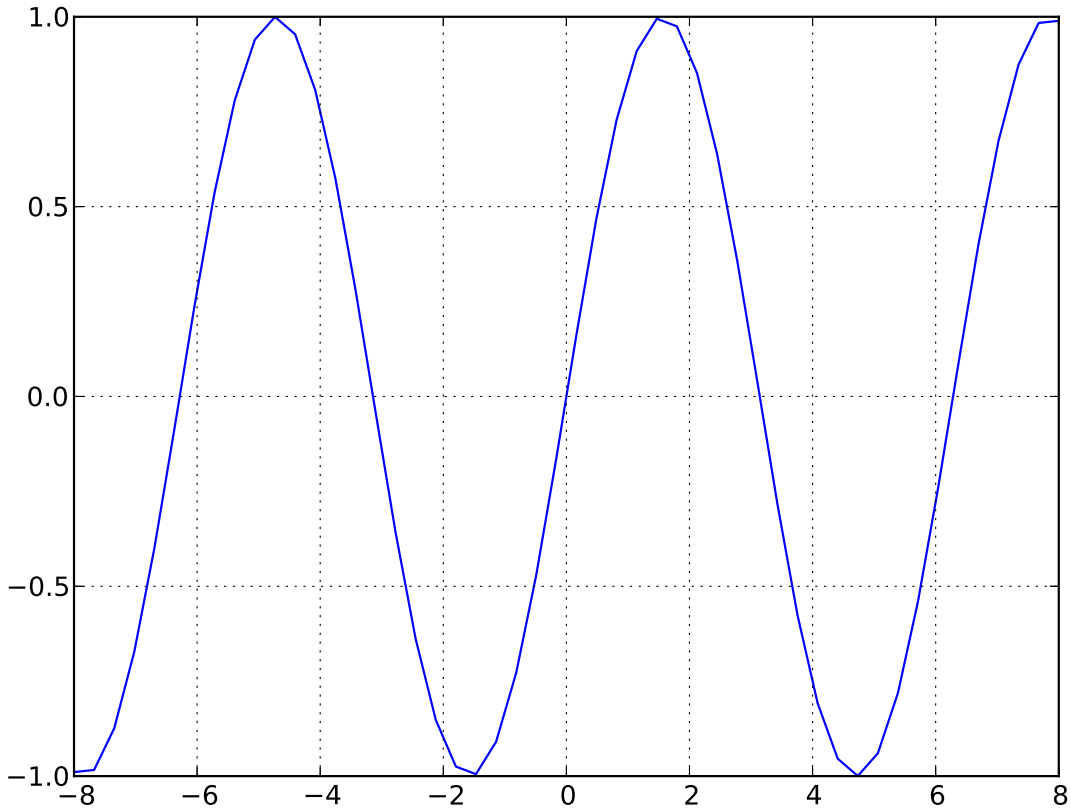


Figura 1: *La función tiene varias raíces en el intervalo.*

Tipo de raíces

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.

Tipo de raíces

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

$$f(x) = a_0\, x^3 + a_1\, x^2 + a_2\, x + a_3$$

1. Tres raíces reales distintas.

Tipo de raíces

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

$$f(x) = a_0\, x^3 + a_1\, x^2 + a_2\, x + a_3$$

1. Tres raíces reales distintas.
2. Una raíz real con multiplicidad 3.

Tipo de raíces

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

$$f(x) = a_0\, x^3 + a_1\, x^2 + a_2\, x + a_3$$

1. Tres raíces reales distintas.
2. Una raíz real con multiplicidad 3.
3. Una raíz real simple y una raíz real con multiplicidad 2.

Tipo de raíces

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

$$f(x) = a_0\, x^3 + a_1\, x^2 + a_2\, x + a_3$$

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

$$f(x) = x^3 - 3\,x^2 - x + 3$$
$$= (x-3)(x+1)(x-1)$$

Las raíces son:

$$x_1 = 3$$
$$x_2 = 1$$
$$x_3 = -1$$

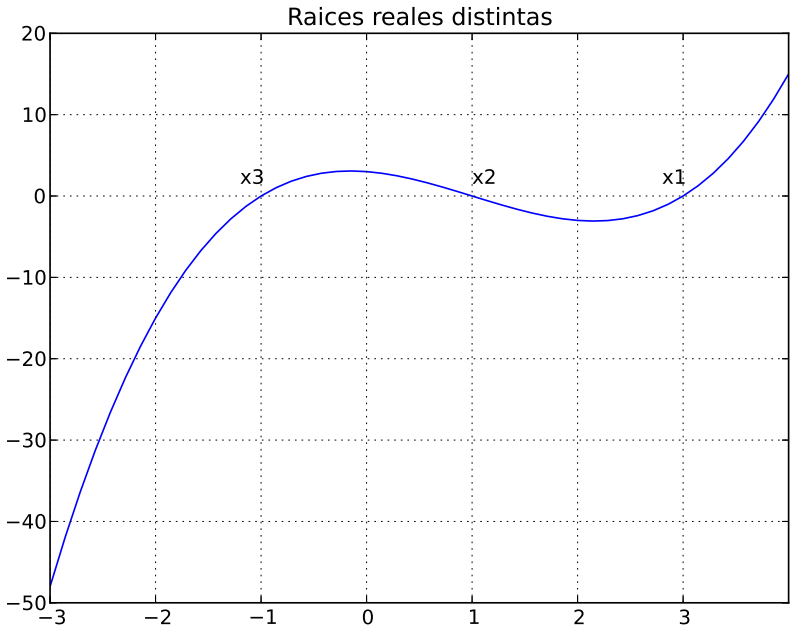


Figura 2: *Los valores x_i representan las raíces del polinomio.*

Raíz real con multiplicidad 3

$$f(x) = x^3 - 6\,x^2 + 12\,x - 8$$
$$= (x-2)^3$$

Las raíces son:

$$x_1 = 2$$
$$x_2 = 2$$
$$x_3 = 2$$

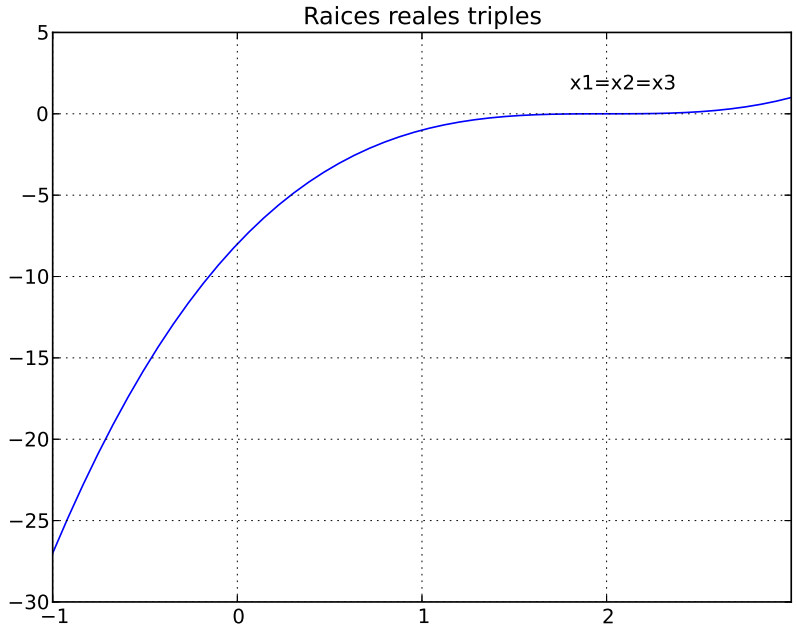


Figura 3: *El valor de las raíces x_i coinciden.*

Raíz real y una raíz real con multiplicidad 2

$$f(x) = x^3 - 12\,x + 16$$
$$= (x+4)(x-2)^2$$

Las raíces son:

$$x_1 = -4$$
$$x_2 = 2$$
$$x_3 = 2$$

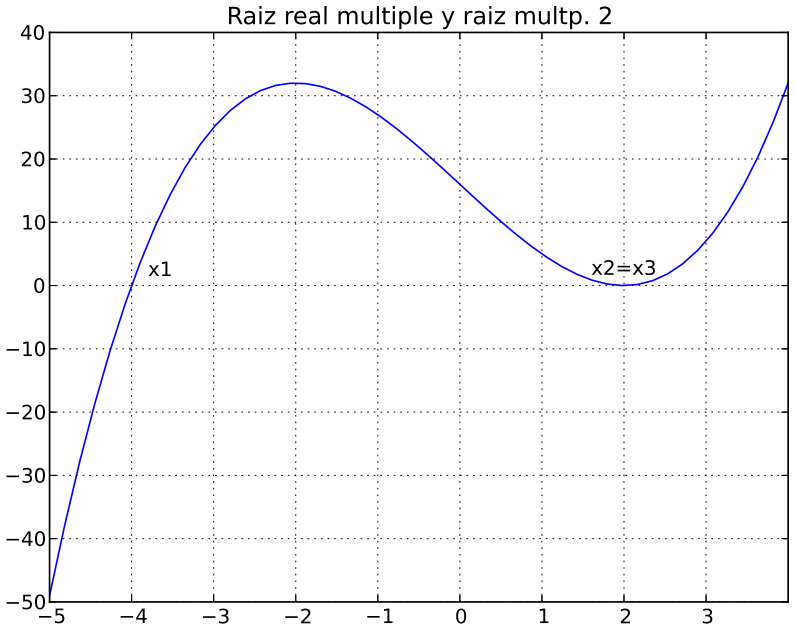


Figura 4: *Dos valores de las raíces coinciden.*

Raíz real y un par conjugado complejo

$$f(x) = x^3 - 2\,x^2 - 3\,x + 10$$
$$= (x+2)(x-(2+i)) *$$
$$*\,(x-(2-i))$$

Las raíces son:

$$x_1 = -2$$
$$x_2 = 2 + i$$
$$x_3 = 2 - i$$

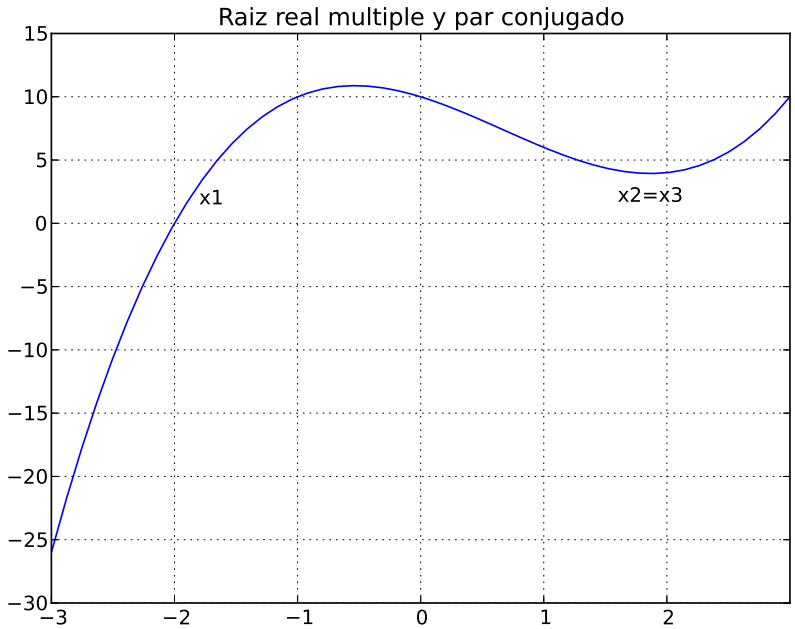


Figura 5: *Dos valores de las raíces son complejos.*

1. Bases generales

2. Funciones algebraicas y trascendentales
   2.1 Funciones algebraicas
   2.2 Funciones trascendentales
   2.3 Encontrar las raíces

3. Método de incrementos sucesivos

4. Métodos para el cálculo de raíces

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.

Funciones algebraicas

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.

Funciones trascedentales

Son aquellas funciones que no son algebraicas.

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

Funciones trascedentales

Son aquellas funciones que no son algebraicas.

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

Ejemplos:

$$f(x) = ln(x^2 - 1)$$
$$g(x) = e^{-0.2\,x}\,\sin(3\,x - 5)$$

Encontrar las raíces

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

1. La determinación de las raíces reales de ecuaciones algebraicas y trascendentales.

Encontrar las raíces

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.

Encontrar las raíces

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

Encontrar las raíces

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.

1. Bases generales

2. Funciones algebraicas y trascendentales

3. Método de incrementos sucesivos

3.1 El método de incrementos sucesivos

3.2 Código Método de incrementos sucesivos

3.3 Ejercicio

4. Métodos para el cálculo de raíces

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.

Método de incrementos sucesivos

El método de búsqueda incremental es una herramienta útil que podemos adoptar en conjunto con otras estrategias de cálculo de raíces.

Éste método no nos ofrece más que una referencia sobre el intervalo en dónde podría(n) estar la(esas) raíces.

Método de incrementos sucesivos

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

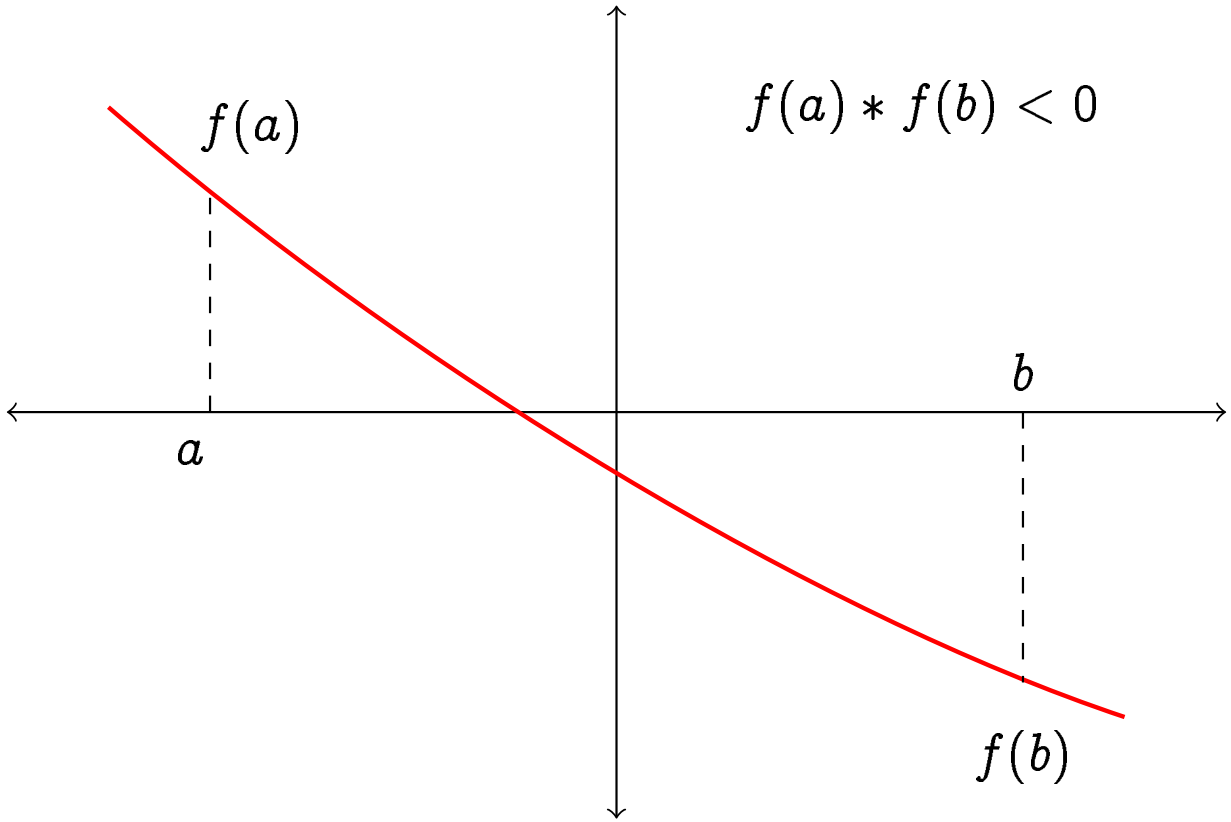


Figura 6: *Los puntos de la función evaluada en los extremos tienen signos contrarios.*

Caso en donde no es posible encontrar la raíz

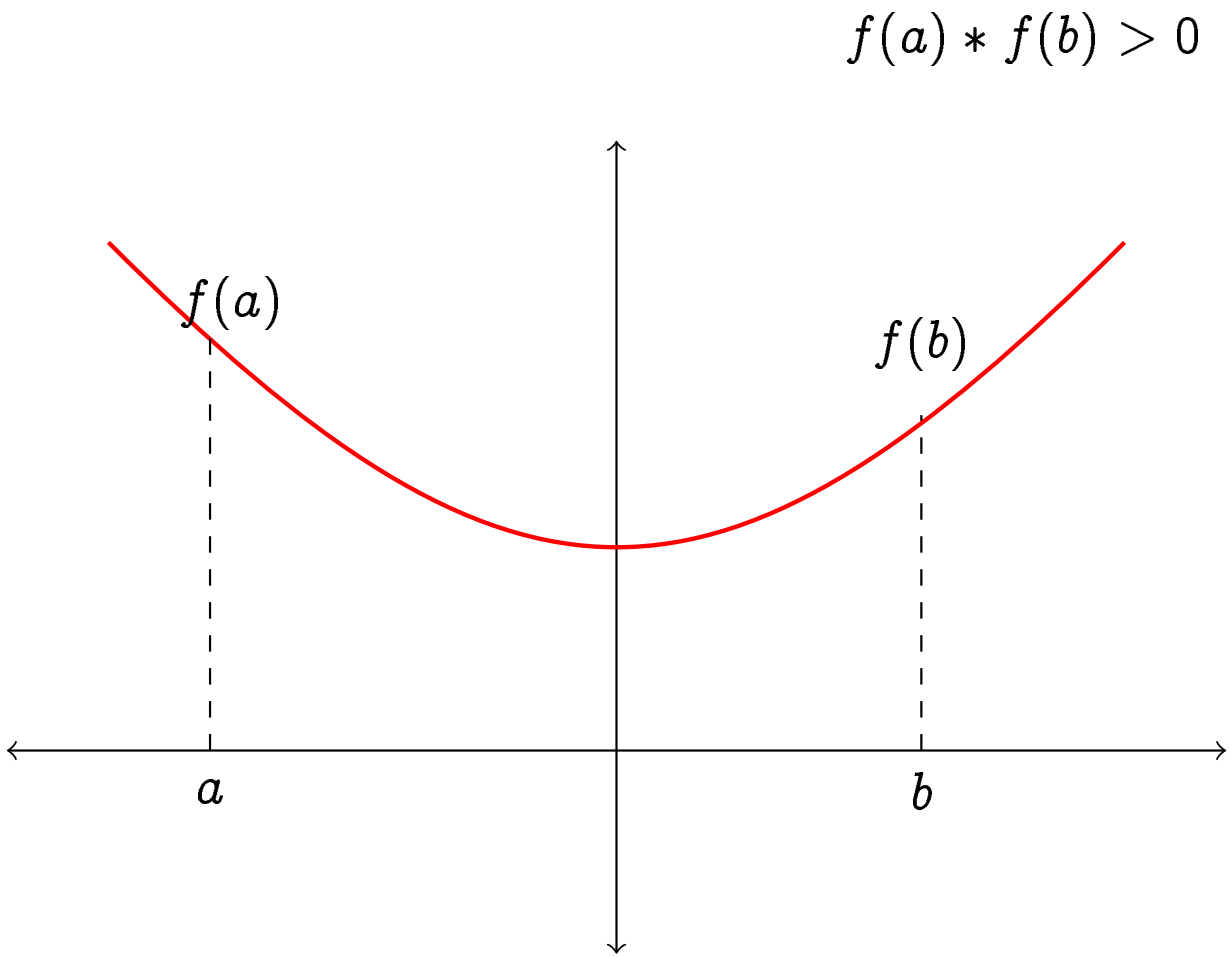


Figura 7: *Los puntos de la función evaluada en los extremos tienen el mismo signo.*

Cuando hay una raíz

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 en intervalos $\Delta\, x$ y mirando cuando se presente un cambio de signo en la función.

Consideraciones con el método

Hay varios problemas con el método de incrementos sucesivos:

1. Es posible perder dos raíces muy próximas entre sí, si el incremento de búsqueda Δx es mayor que la separación de las raíces.

Consideraciones con el método

Hay varios problemas con el método de incrementos sucesivos:

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.

Consideraciones con el método

Hay varios problemas con el método de incrementos sucesivos:

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 $n = 1, 3, 5, \ldots$

Casos en donde no hay una raíz

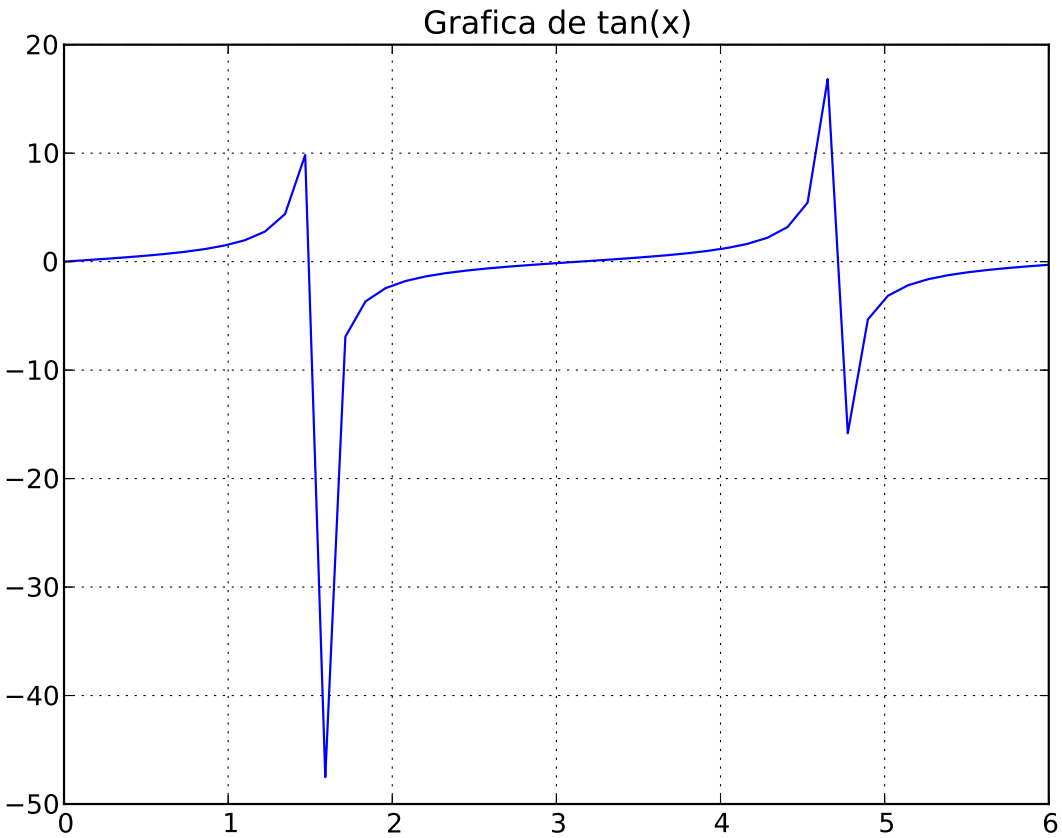


Figura 8: *Estos puntos no son ceros verdaderos, ya que la función no cruza el eje* x.

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.

Código Método de incrementos sucesivos

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.

Función **buscaraiz**

Código 1: Función buscaraiz

```
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 para encontrar una raíz

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

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

¿Cómo resolver el problema?

Como el ejercicio pide que se localice la raíz positiva más pequeña, habrá que tener una idea sobre el comportamiento de la función.

Para ello, graficamos en un intervalo inicial, considera que hay potencias al cubo.

Gráfica inicial de la función

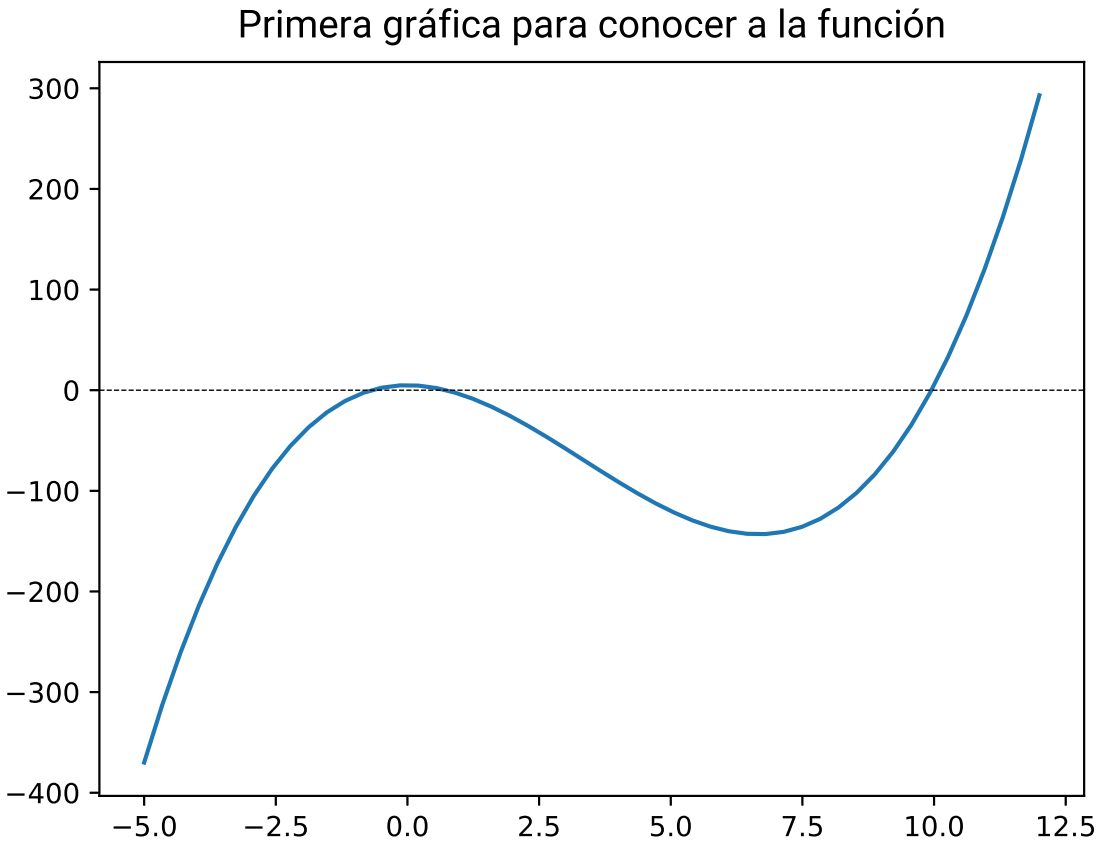


Figura 9: *De la gráfica vemos que hay tres raíces, que cruzan el eje* $y = 0$.

Acotando el problema

Como se nos pide el valor de la raíz positiva más pequeña, descartamos las raíz negativa y la otra raíz positiva.
En la siguiente gráfica se reduce el intervalo sobre el eje x para tener una mejor vista.

El intervalo de la gráfica se ha acotado

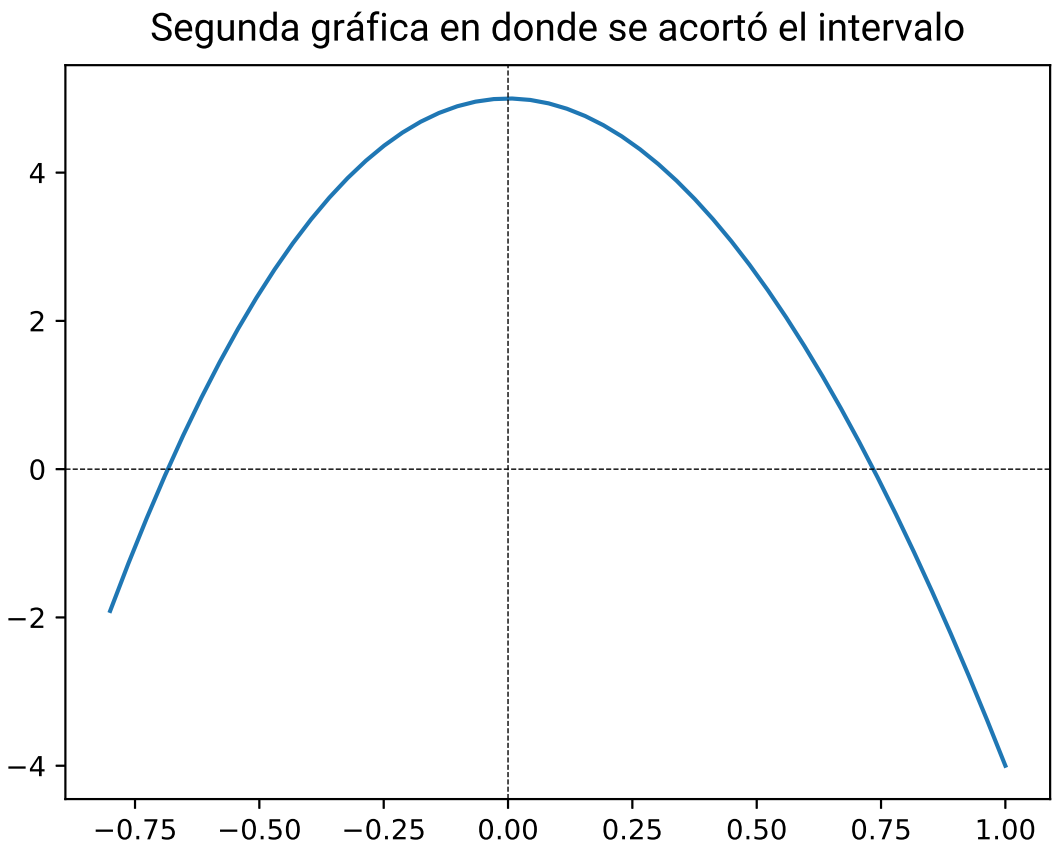


Figura 10: *Se aprecia mucho mejor al reducir el intervalo, pero nos falta aún estimar con el código el intervalo donde está la raíz positiva más pequeña.*

Ahora con python

Código 2: Solución con python

```
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
```

El intervalo de trabajo

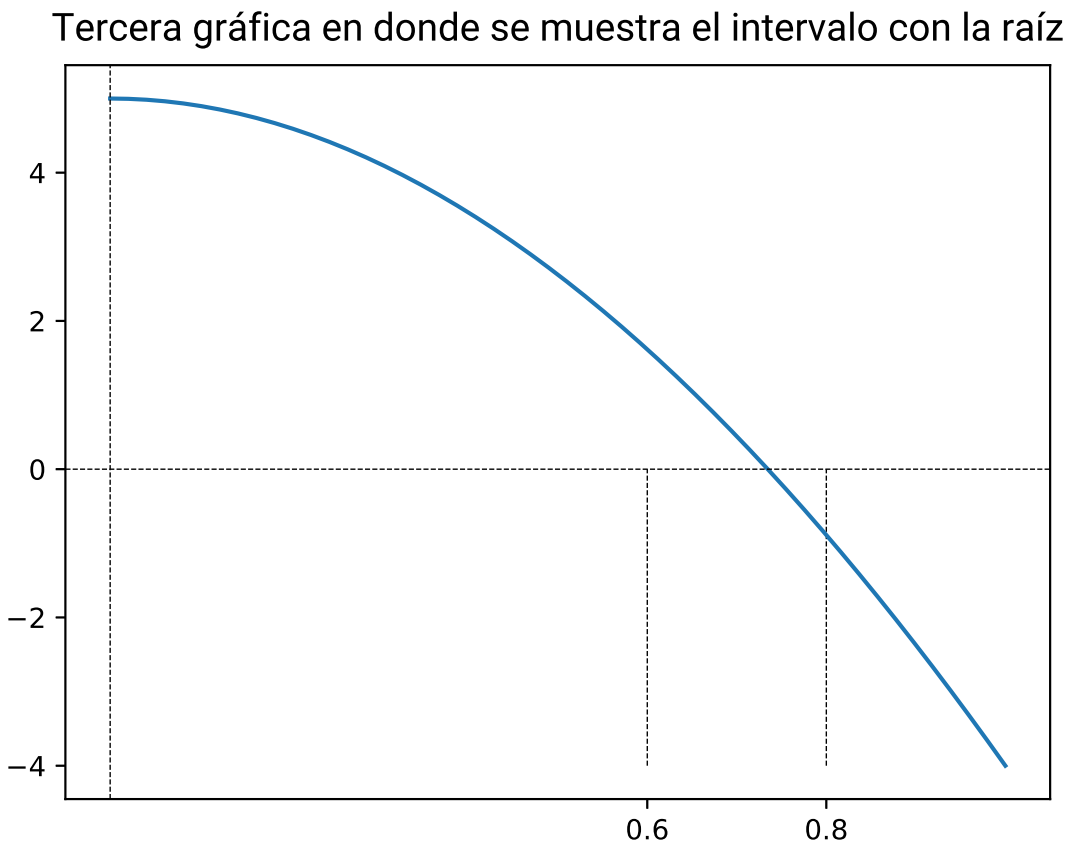


Figura 11: *El código devuelve el intervalo de interés, por lo que ahora hay que usar alguna técnica para conocer el valor de la raíz.*

1. Bases generales

2. Funciones algebraicas y trascendentales

3. Método de incrementos sucesivos

4. Métodos para el cálculo de raíces

4.1 Método de Bisección

4.2 Método de Newton-Raphson

4.3 Método de la falsa posición

4.4 Método de la falsa posición modificado

4.5 Método de la secante

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.

Método de Bisección

El método de bisección logra esta tarea: el intervalo se reduce sucesivamente a la mitad hasta que se vuelve suficientemente pequeño.

Método de Bisección

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.

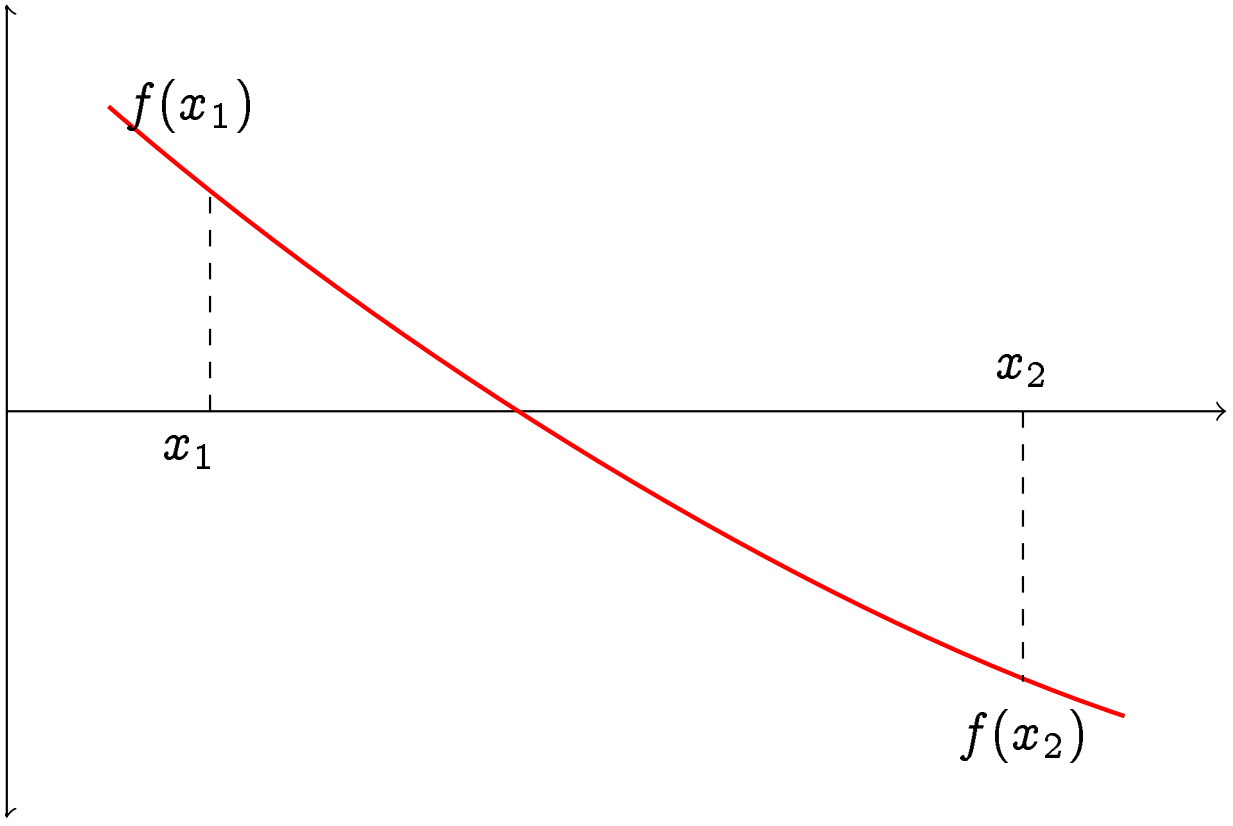
Funcionamiento del método

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

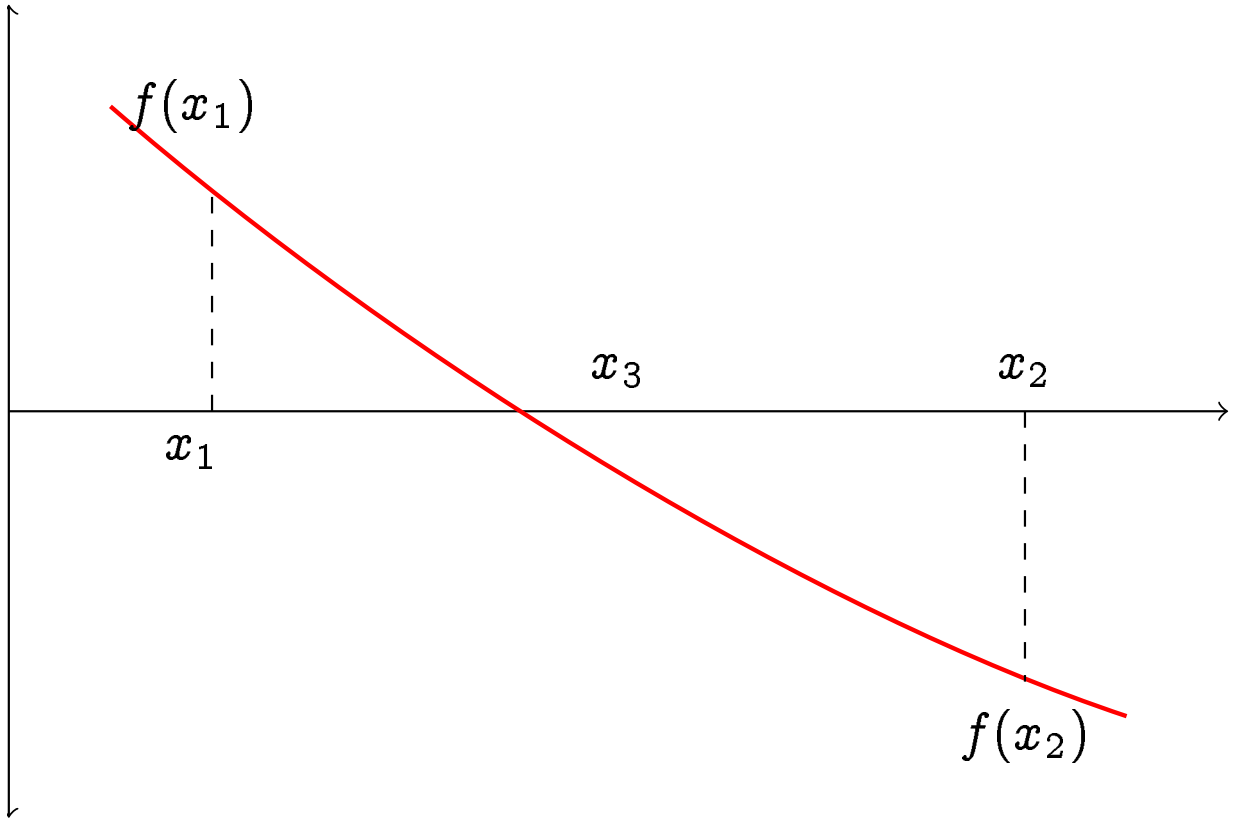
Funcionamiento del método

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.

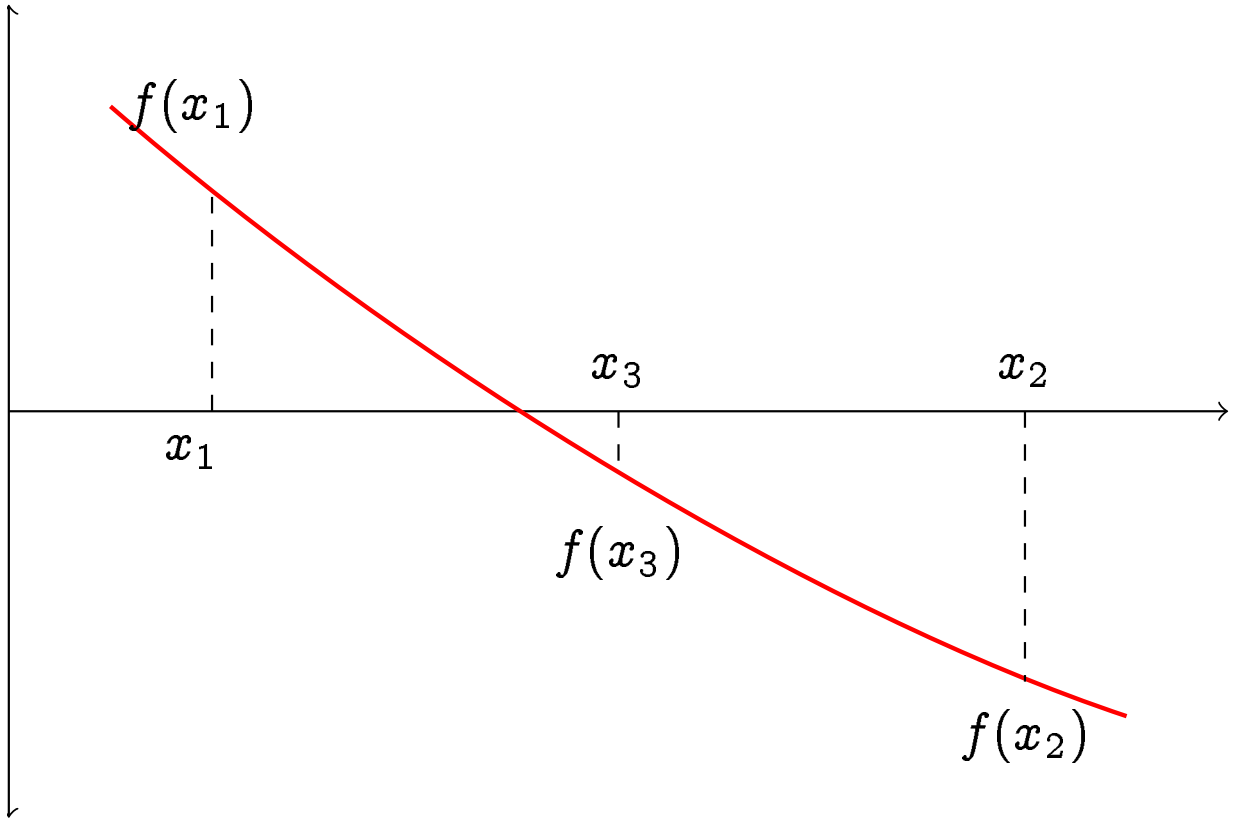
Descripción gráfica del método de bisección



Descripción gráfica del método de bisección



Descripción gráfica del método de bisección

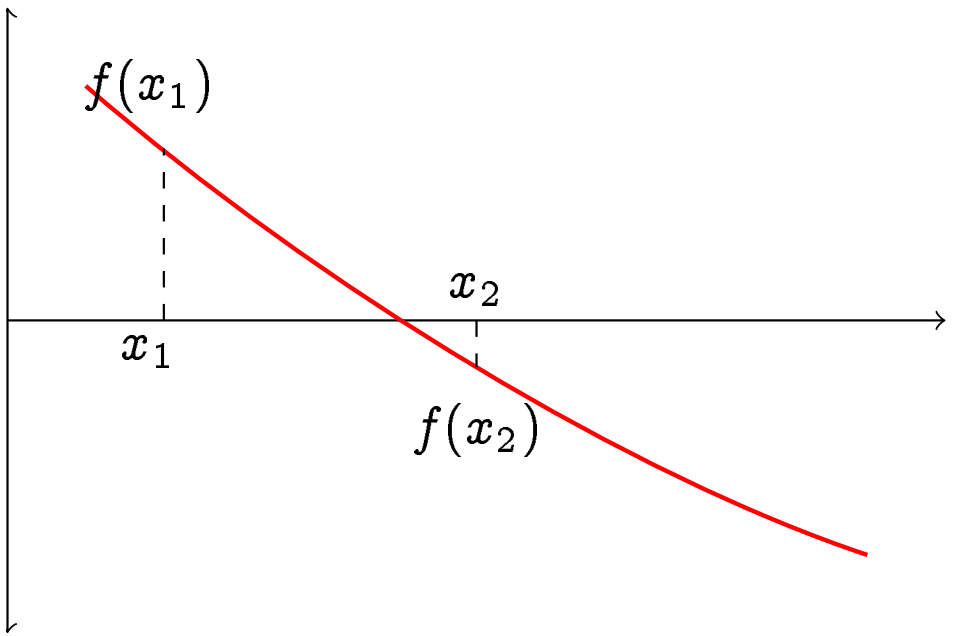


Funcionamiento del método

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, y se repite la división del intervalo.

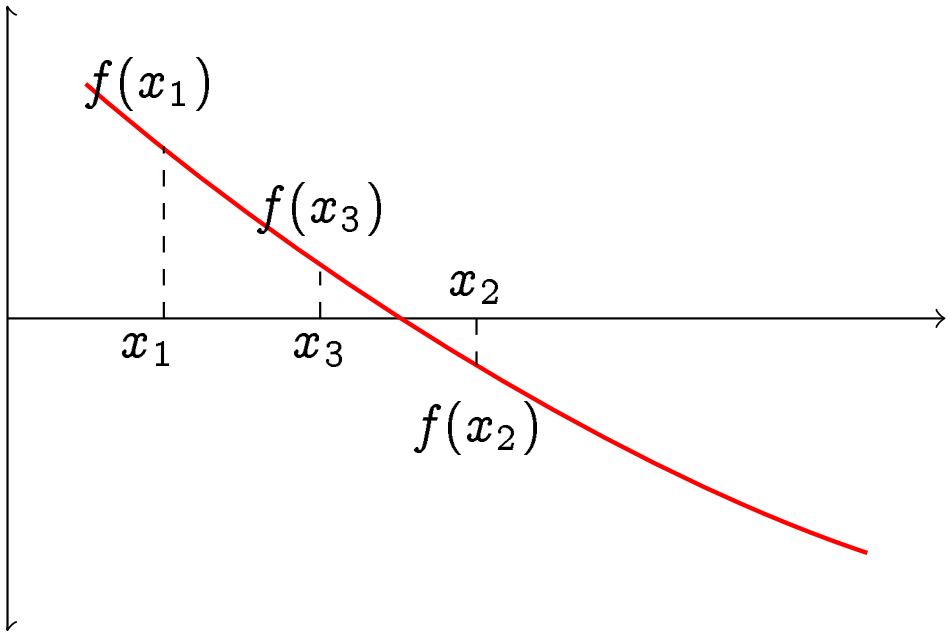
Funcionamiento del método

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, y se repite la división del intervalo.



Funcionamiento del método

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, y se repite la división del intervalo.



Funcionamiento del método

De lo contrario, la raíz se encuentra en el intervalo (x_3, x_2), en tal caso, se sustituye x_3 por x_1, y se repite la división del intervalo.

¿Hasta cuando se repite la división del intervalo?

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

$$|x_2 - x_1| \leq \varepsilon$$

Estimar el número de divisiones del intervalo

Es fácil calcular el número de biseccciones necesarias para alcanzar el valor de ε.

Estimar el número de divisiones del intervalo

Es fácil calcular el número de biseccciones necesarias para alcanzar el valor de ε.

El intervalo inicial $\Delta\, x$, se reduce a $\Delta\, x/2$ en la primera bisección, $\Delta\, x/2^2$ en la segunda, luego de n biseccciones, $\Delta\, x/2^n$.

Estimar el número de divisiones del intervalo

Es fácil calcular el número de biseccciones necesarias para alcanzar el valor de ε.

El intervalo inicial $\Delta\, x$, 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 $\Delta\, x/2^n = \varepsilon$, resolvemos para n

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

Función el método de bisección I

Código 3: Método de bisección con `python`

```
1 def biseccion(f, x1, x2, switch = 1, tol
      = 1.0e-9):
2     f1 = f(x1)
3     if f1 == 0.0: return x1
4
5     f2 = f(x2)
6     if f2 == 0.0: return x2
7
8     if sign(f1) == sign(f2):
9         print('La raiz no esta en el
    intervalo')
```

Función el método de bisección II

```
10
11      n = int(math.ceil(math.log(abs(x2 - x1)/tol)/math.log(2.0)))
12
13      for i in range(n):
14          x3 = 0.5 * (x1 + x2); f3 = f(x3)
15          if (switch == 1) and (abs(f3) > abs(f1)) \
16                          and (abs(f3) > abs(f2)):
17
18              return None
19
```

Función el método de bisección III

```
20          if f3 == 0.0: return x3
21
22          if sign(f2) != sign(f3): x1 = x
    3; f1 = f3
23          else: x2 = x3; f2 = f3
24
25      return (x1 + x2)/2.0
```

El argumento `switch`

Al establecer el valor del argumento `switch` $= 1$, forzamos la rutina para verificar si la magnitud de $f(x)$ disminuye con cada intervalo a la mitad.

El argumento `switch`

Si no lo hace, algo puede estar mal: probablemente la “raíz”, no es una raíz en absoluto, pero un polo.

En ese caso, se devuelve `raíz = None`. Porque esta característica no siempre es deseable, el valor predeterminado es `switch` $= 0$.

¿Qué hace la función `ceil`?

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
```

Ejemplo

Calcular la raíz de

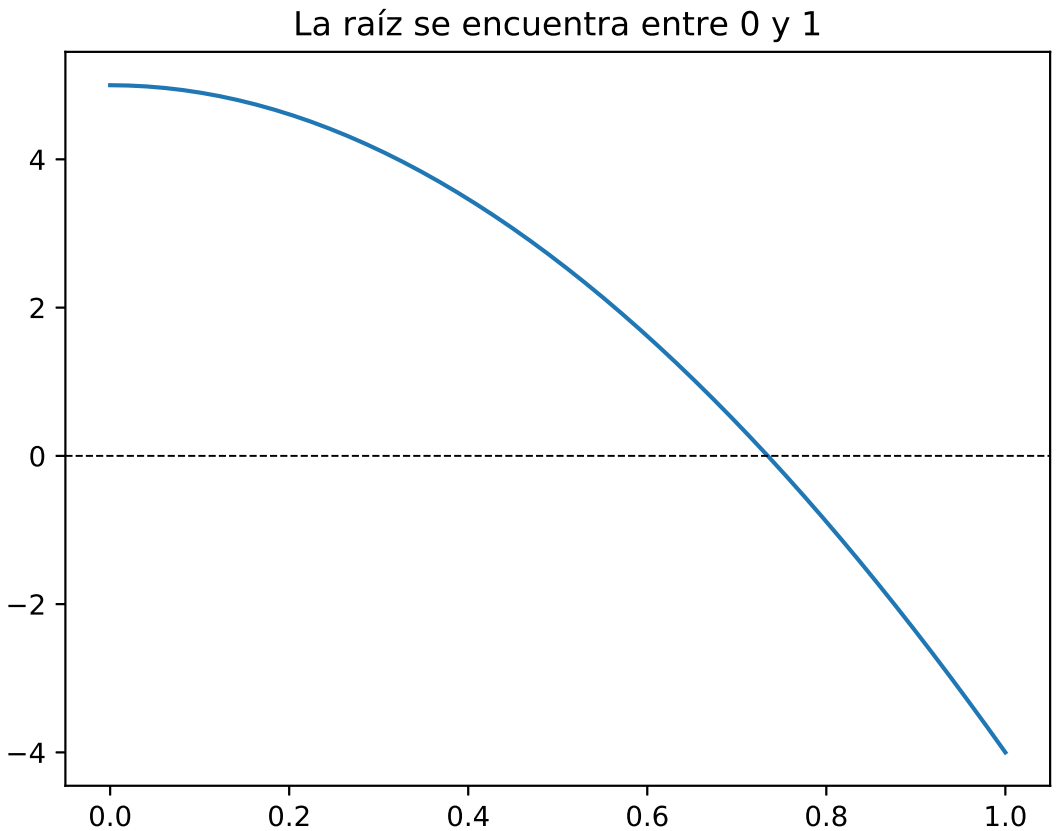
$$x^3 - 10\,x^2 + 5 = 0$$

que se encuentra en el intervalo $(0, 1)$, con una precisión de 4 dígitos.

¿Cuántas evaluaciones se requieren para encontrar la raíz?

Ejemplo

Como primer punto, generamos una gráfica para tener una idea del comportamiento de la función



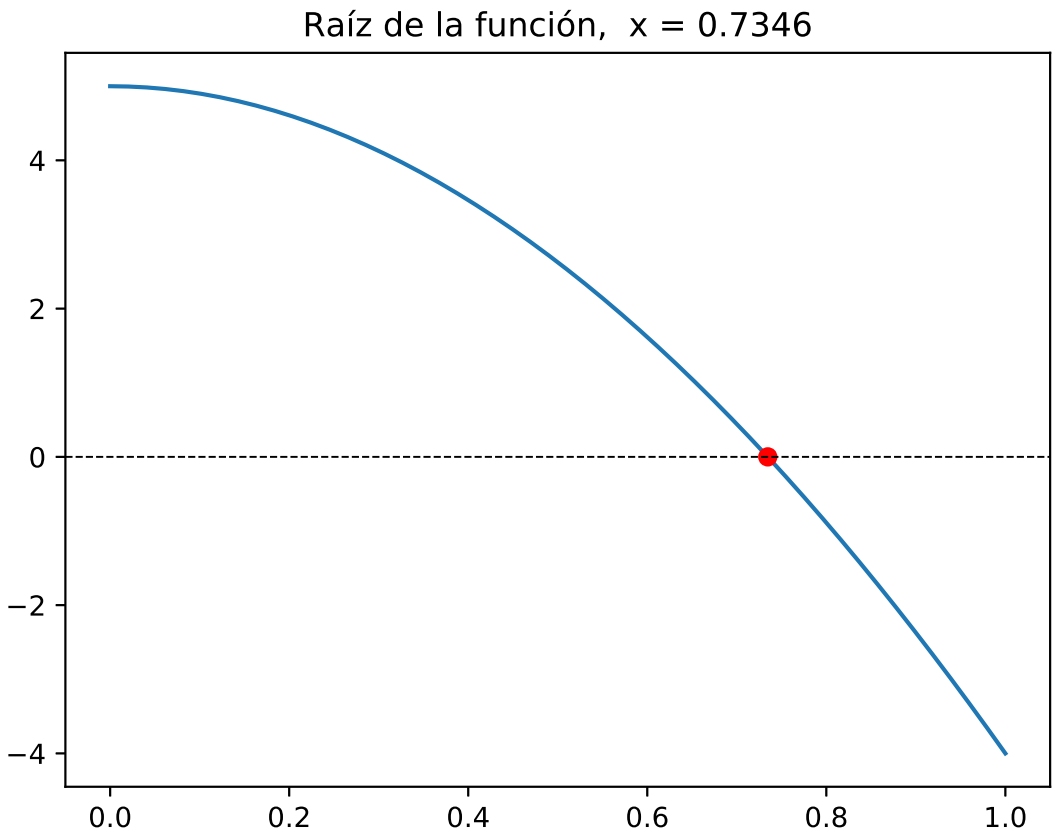
Solución

Código 4: Solución al ejercicio

```
1 from ModuloRaices import biseccion
2
3 def f(x): return x**3 - 10.0*x**2 + 5.0
4
5 raiz = biseccion(f, 0., 1.0, tol = 1.0e-
    4)
6
7 print('raiz=', '{:6.4f}'.format(raiz))
```

Solución gráfica

Una vez calculada la raíz, podemos presentarla en la gráfica



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.

Método de Newton-Raphson

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:

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

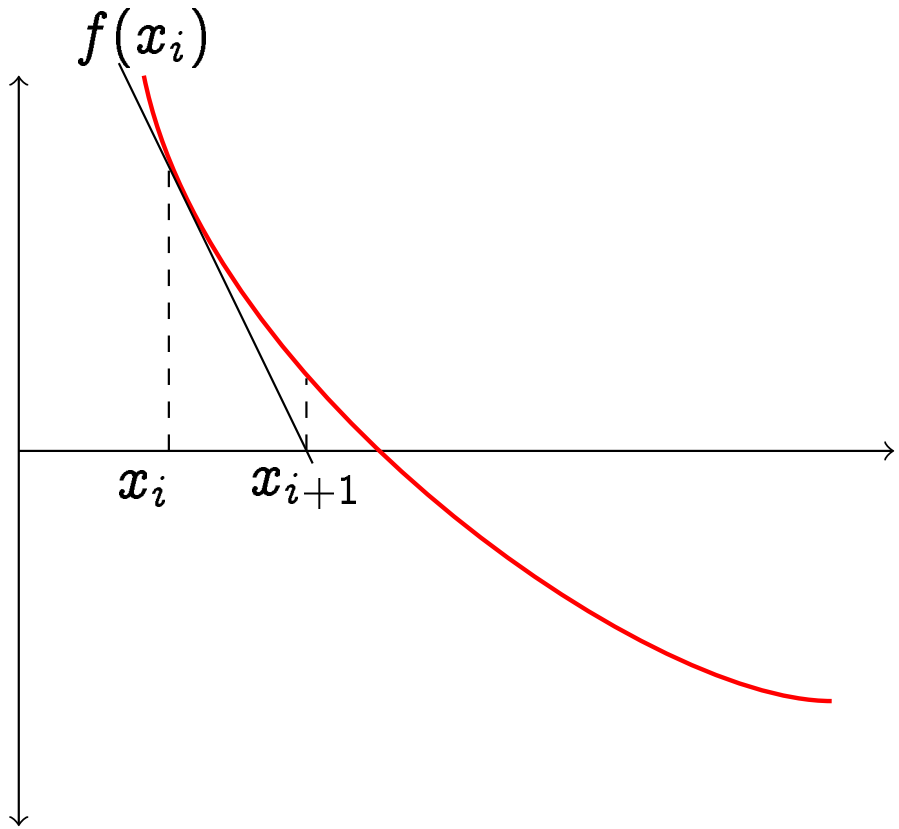
Si x representa el valor verdadero de la raíz, el error en x_i es $E_i = x - x_i$. Se puede demostrar que si x_{i+1} se calcula de la expresión de N-R, el error es:

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

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:

$$\left|x_{i+1} - x_i\right| < \varepsilon$$

El algoritmo es el siguiente:

1. Sea x un valor inicial para la raíz de $f(x) = 0$.

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:

$$\left|x_{i+1} - x_i\right| < \varepsilon$$

El algoritmo es el siguiente:

1. Sea x un valor inicial para la raíz de $f(x) = 0$.
2. Calcular $\Delta x = -f(x)/f'(x)$.

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:

$$\left|x_{i+1} - x_i\right| < \varepsilon$$

El algoritmo es el siguiente:

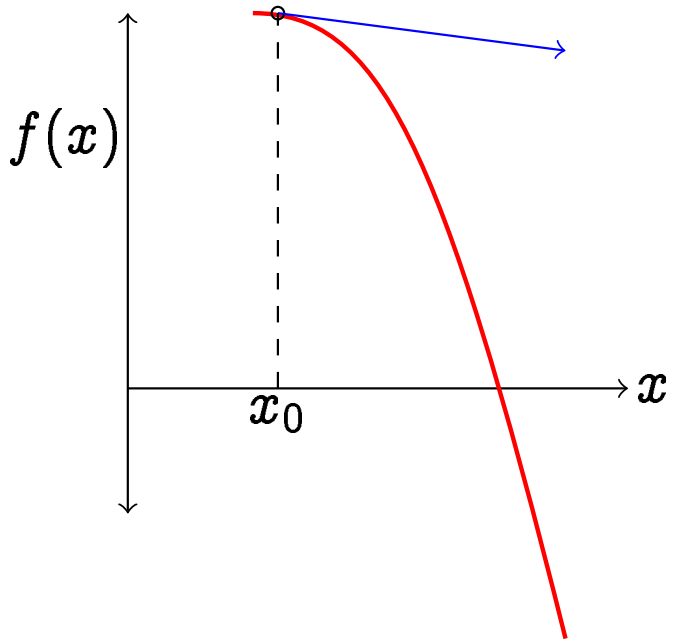
1. Sea x un valor inicial para la raíz de $f(x) = 0$.
2. Calcular $\Delta x = -f(x)/f'(x)$.
3. Asignar $x \leftarrow x + \Delta x$ y se repiten los pasos 2-3, hasta alcanzar $\left|\Delta x\right| < \varepsilon$.

Precaución con el método N-R

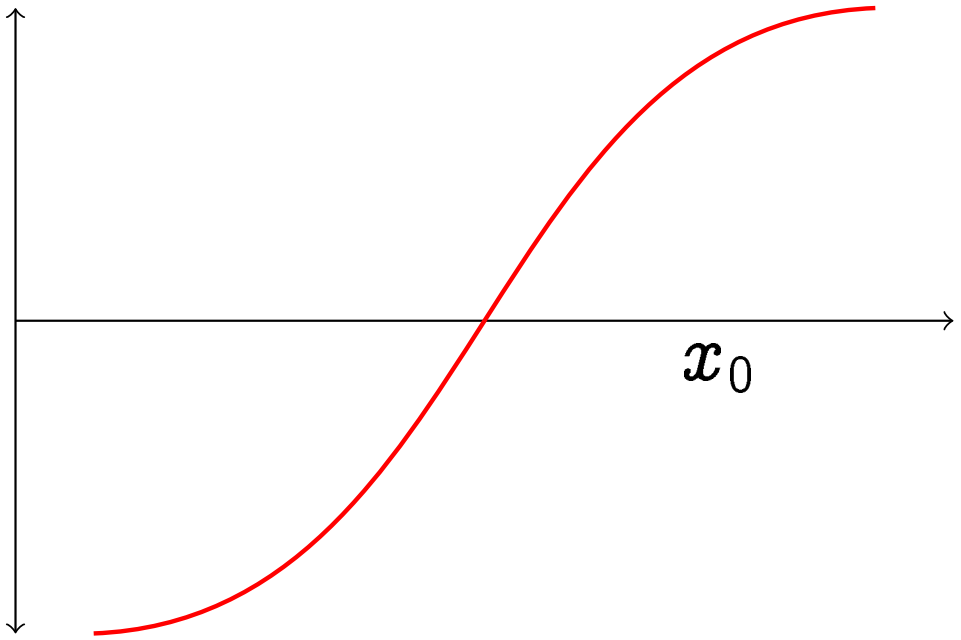
Aunque el método de Newton-Raphson converge rápidamente cerca de la raíz, sus características globales de convergencia son pobres.

La razón es que la línea tangente no es siempre una aproximación aceptable de la función.

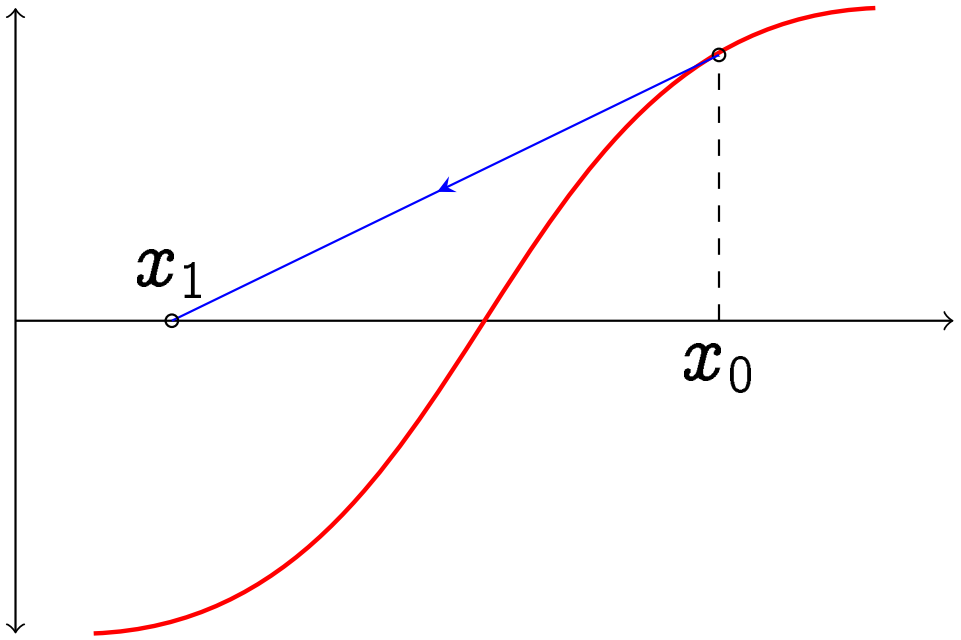
Precaución con el método N-R



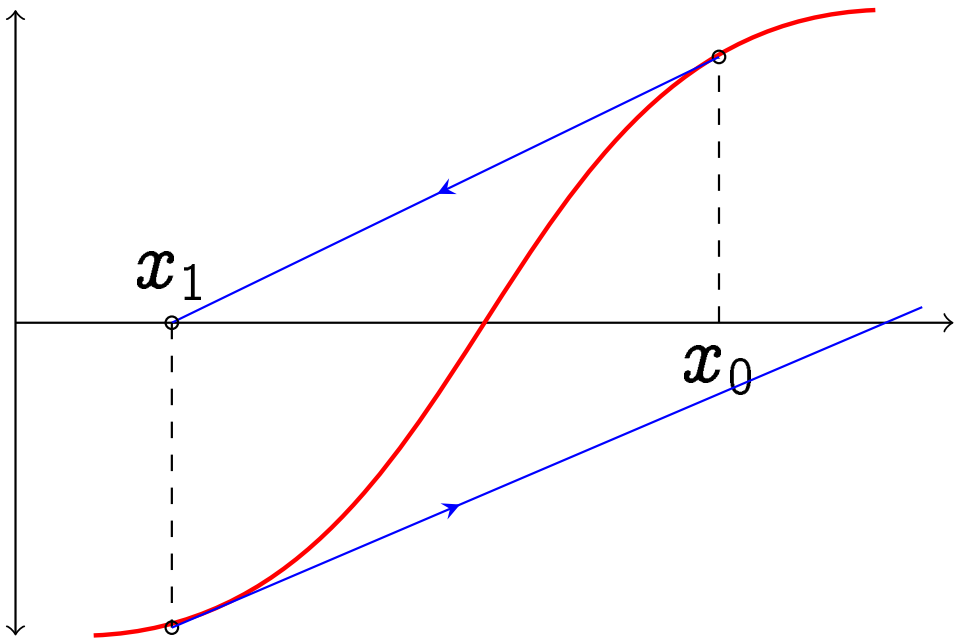
Precaución con el método N-R



Precaución con el método N-R



Precaución con el método N-R



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).

Algoritmo para el método Newton-Raphson

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.

Algoritmo para el método Newton-Raphson

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 I

Código 5: Código del método N-R

```
1 def newtonRaphson(f, df, a, b, tol = 1.0
    e-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)
8
9     for i in range(30):
```

Algoritmo de Newton-Raphson II

```
10          fx = f(x)
11          if abs(fx) < tol: return x
12
13          if fa*fx < 0.0:
14              b = x
15          else:
16              a = x; fa = fx
17
18          dfx = df(x)
19
20          try: dx = -fx/dfx
21          except ZeroDivisionError: dx = b
    - a
```

Algoritmo de Newton-Raphson III

```
22            x =   x + dx
23
24            if(b - x)*(x - a) < 0.0:
25                dx = 0.5*(b-a)
26                x = a + dx
27
28            if abs(dx) < tol*max(abs(b),1.0)
    : return x
29
30      print 'Son demasiadas iteraciones'
```

Primer ejercicio

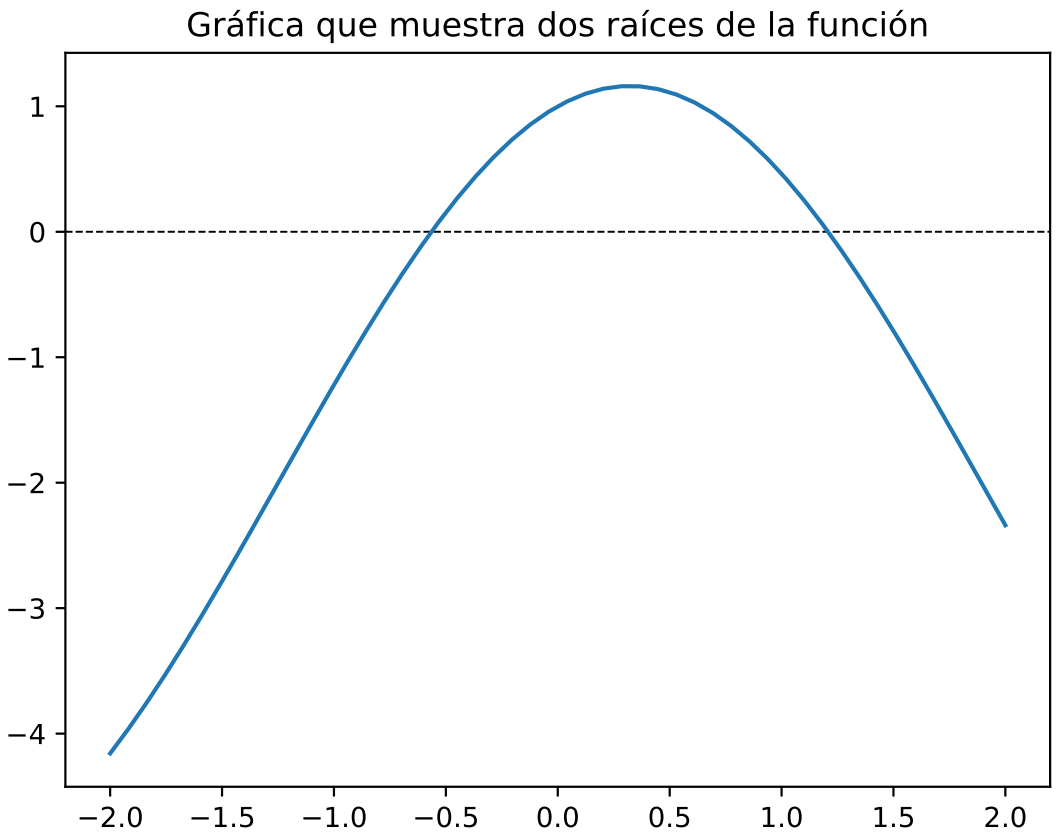
Usando el método de Newton-Raphson, calcular las dos raíces de la función

$$\sin\ x + 3\ \cos\ x - 2 = 0$$

en el intervalo $(-2, 2)$

Primer paso: graficar la función

La gráfica de la función en el intervalo dado es la siguiente:



Intervalos de las raíces

Con el método de incrementos sucesivos, estimamos cada intervalo donde se encuentran las raíces de la función.

Intervalos de las raíces

Código 6: Intervalos para las raíces

```
inicio, final, dx = -2., 2. , 0.01

a!1!, b!1! = buscaraiz(f, inicio, final,
    0.1)
print('Una raiz esta en  el intervalo ['
   , a!1!, ',', b!1! , ']')

inicio = b!1!

a!2!, b!2! = buscaraiz(f, inicio, final,
    0.1)
print('Una raiz esta en  el intervalo ['
   , a!2!, ',', b!2! , ']')
```

Los intervalos

El resultado que nos devuelve la función de incrementos sucesivos es:

```
Una raíz está en [ -0.6 , -0.5 ]
Una raíz está en [ 1.2 , 1.3 ]
```

Código completo I

Código 7: Intervalos para las raíces

```
1 from ModuloRaices import buscaraiz,
    newtonRaphson
2 import matplotlib.pyplot as plt
3 import numpy as np
4
5
6 def f(x): return np.sin(x) + 3 * np.cos(
    x) - 2
7 def df(x): return np.cos(x) - 3 * np.sin
    (x)
8
```

Código completo II

```
9
10 intervalo!1! = []
11 intervalo!2! = []
12
13 inicio, final, dx = -2., 2. , 0.01
14
15
16 a!1!, b!1! = buscaraiz(f, inicio, final,
       0.1)
17 print('Una raiz est en [', round(a!1!, 2
     ), ',', round(b!1!, 2) , ']')
18 intervalo!1!.append(a!1!)
19 intervalo!1!.append(b!1!)
```

Código completo III

```
20
21 inicio = b!1!
22 a!2!, b!2! = buscaraiz(f, inicio, final,
       0.1)
23 print('Una raiz esta en [', round(a!2!,2
     ), ',', round(b!2!,2), ']')
24 intervalo!2!.append(a!2!)
25 intervalo!2!.append(b!2!)
26
27 raiz!1! = newtonRaphson(f, df, intervalo
     !1![0], intervalo!1![-1], 1.e-4)
28 print(raiz!1!)
29
```

Código completo IV

```
30 raiz!2! = newtonRaphson(f, df, intervalo
     !2![0], intervalo!2![-1], 1.e-4)
31 print(raiz!2!)
32
33 x = np.linspace(-2,2)
34 plt.axhline(y=0, ls='dashed', lw=0.7,
     color='k')
35 plt.plot(x, f(x), color='g')
36 plt.plot(raiz!1!, 0, 'ro', label='Raiz !
     1! = ' + str(round(raiz!1!, 3)))
37 plt.plot(raiz!2!, 0, 'bo', label='Raiz !
     2! = ' + str(round(raiz!2!, 3)))
38 plt.legend(loc='lower right')
```

Código completo V

```
39 plt.title('Grafica que muestra dos
      raices de la funcion')
40 plt.show()
```

Solución

La gráfica de la función en el intervalo dado es la siguiente:

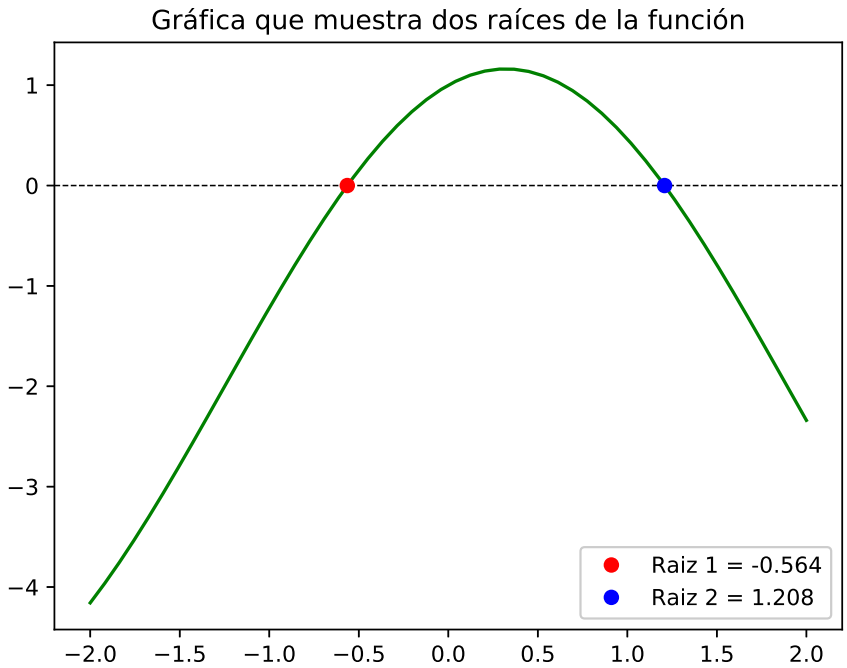


Figura 12: *Implementando varias etapas, se calcularon las raíces de la función en el intervalo.*

Ejercicio

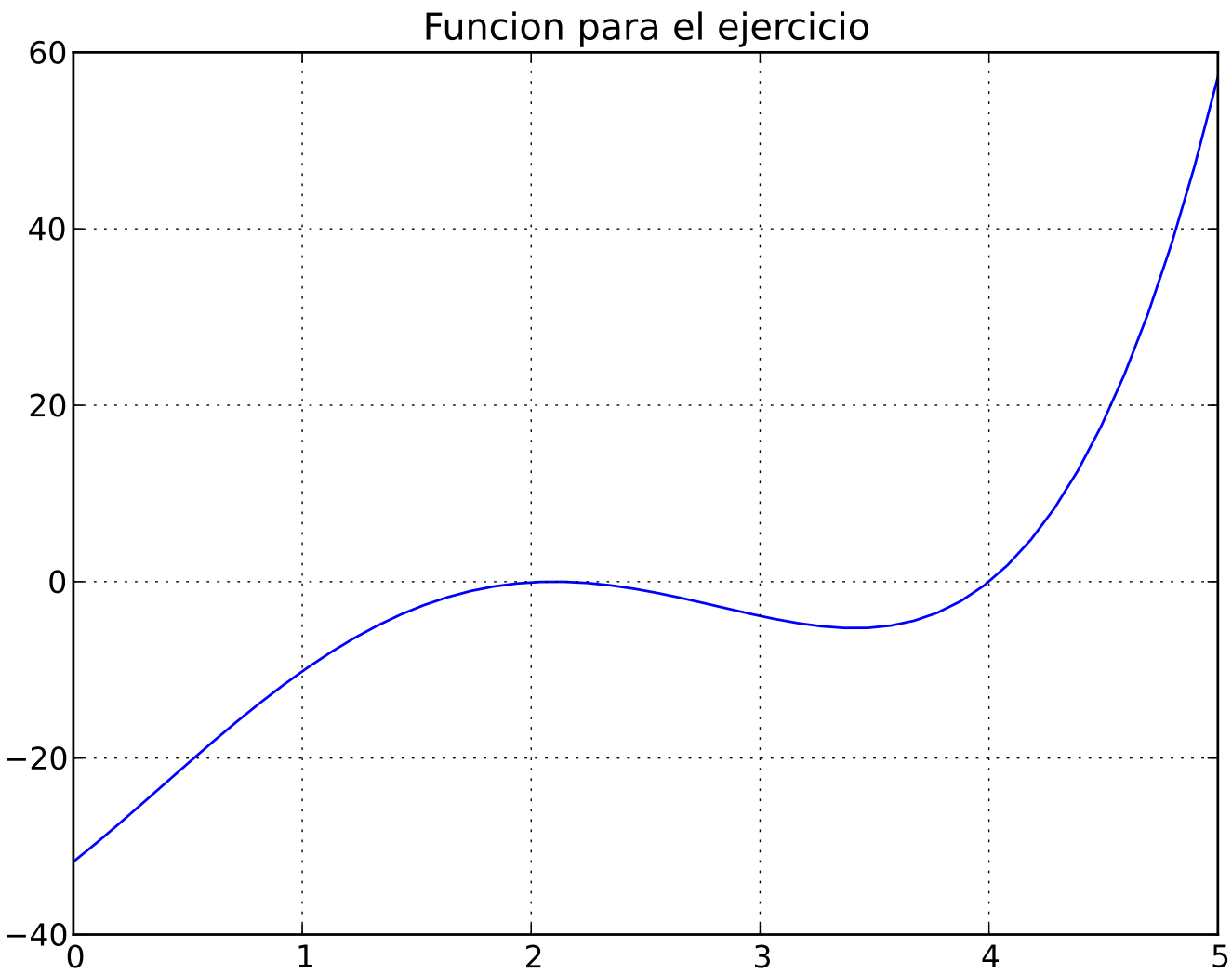
Encontrar la raíz positiva más pequeña de

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

Como en los ejercicios anteriores, es buena idea generar una gráfica de la función.

Gráfica del ejercicio

Gráfica del ejercicio



Caso particular para esta función

Al realizar un poco de álgebra, nos encontramos con lo siguiente:

$$0.002(x-4.)(5.x+9.)(10.x-21.)^2$$

La función tiene dos raíces reales y una de multiplicidad 2.

Ajuste al código de NR

Como el método de NR se basa en detectar el cambio de signo en la función, en este caso, para una raíz con multiplicidad 2, haremos un ajuste al código del método.

Ejercicio I

Código 8: Código para el ejercicio

```
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-09):
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'
```

Ejercicio II

```
10
11 raiz,numIter = newtonRaphson(2.0)
12
13 print 'Raiz =',raiz
14 print 'Numero de iteraciones =',numIter
```

Convergencia del método

Se puede demostrar que la convergencia del método NR cerca de una raíz con multiplicidad es linear, más que cuadrática.

Convergencia del método

Se puede demostrar que la convergencia del método NR cerca de una raíz con multiplicidad es linear, más que cuadrática.

La convergencia de una raíz múltiple se puede acelerar al cambiar la fórmula de NR.

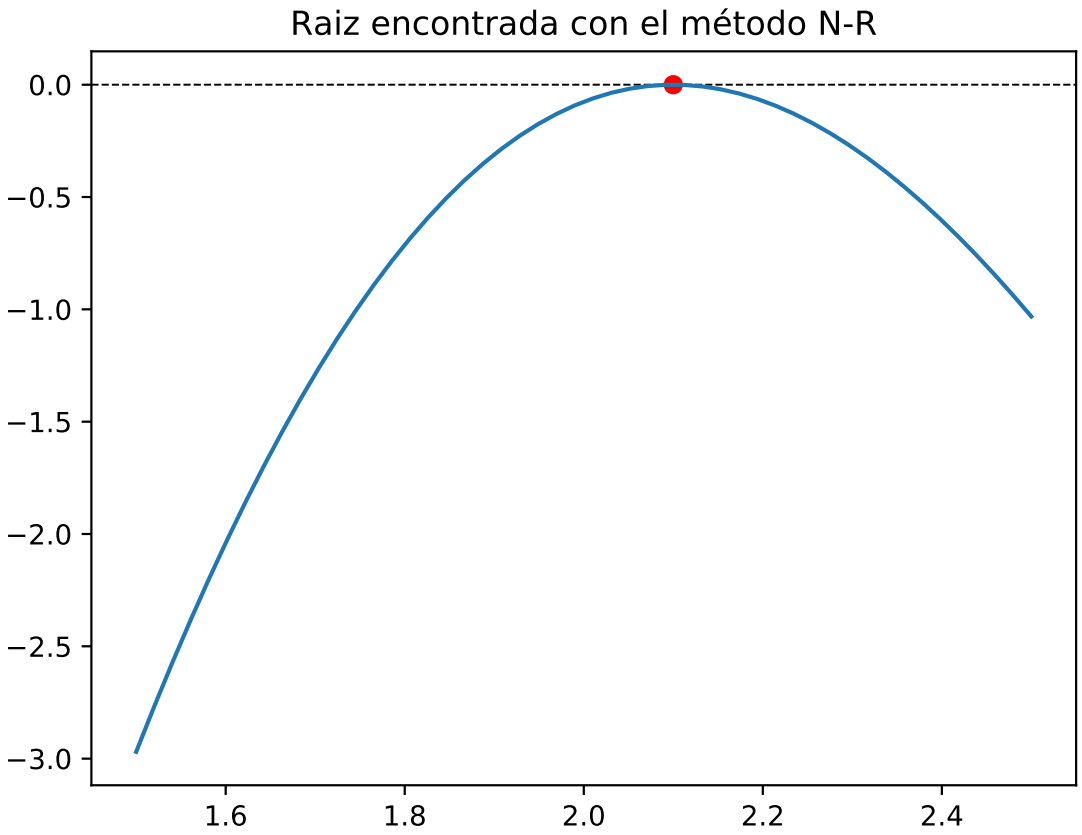
Ajuste a la función NR

Se ajusta de la forma

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

donde m es la multiplicidad de la raíz (en el ejemplo, $m = 2$). Después de hacer los ajustes, el resultado se obtiene en 5 iteraciones.

Solución del ejercicio



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:

$$x = a + \frac{c - a}{f(c) - f(a)}\left[y - f(a)\right]$$

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

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

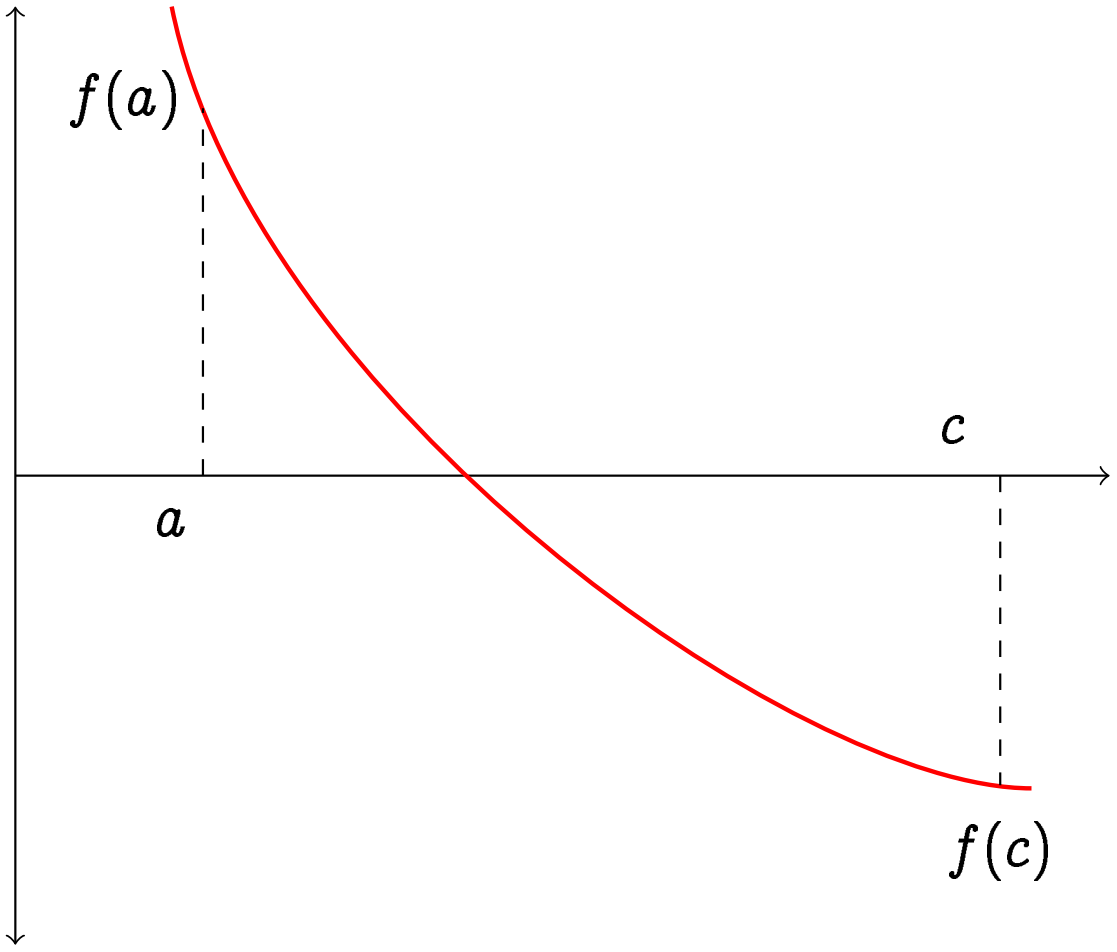
Después de encontrar b, el intervalo $[a, c]$ se divide en $[a, b]$ y $[b, c]$.

Si $f(a)\ f(b) \leq 0$, la raíz se encuentra en $[a, b]$; en caso contrario, está en $[b, c]$.

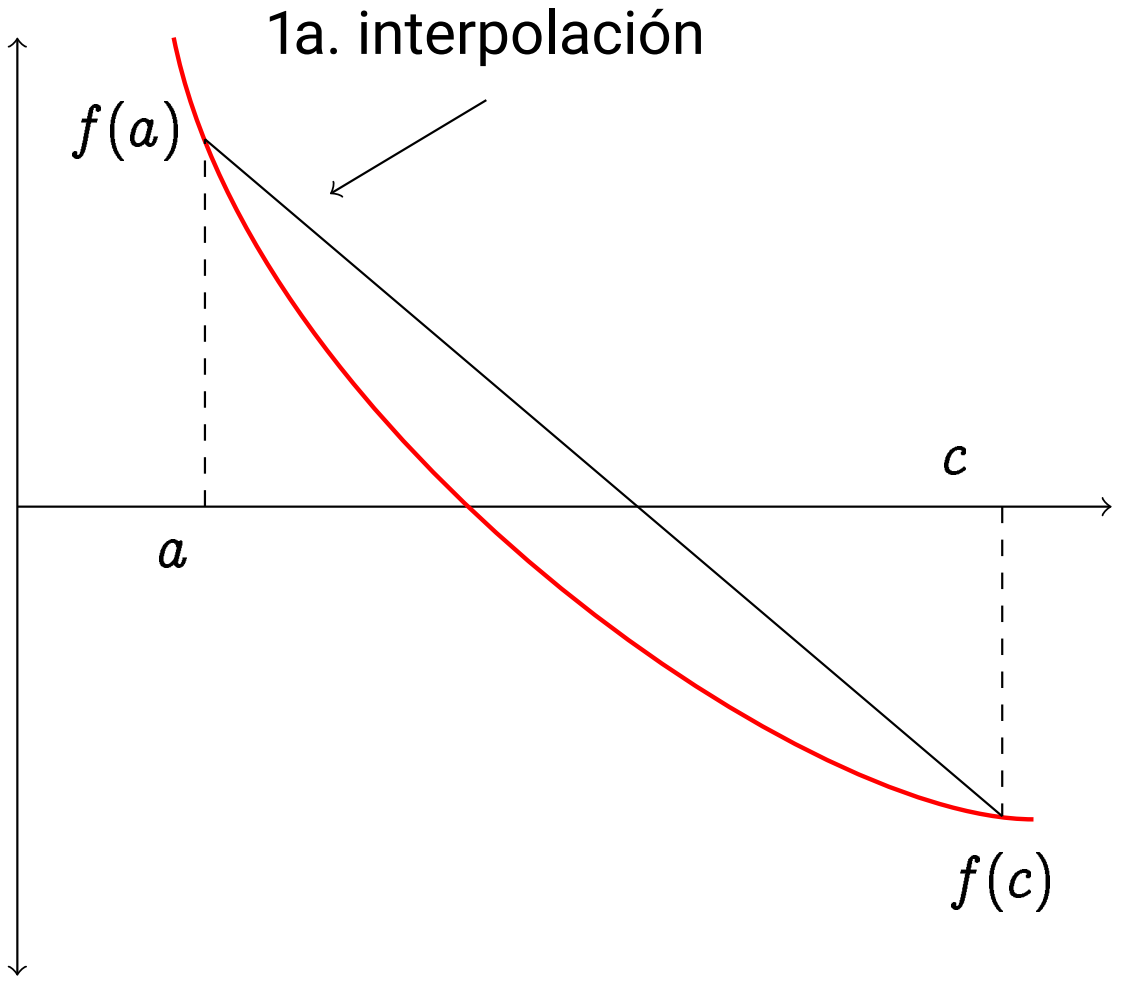
Los extremos del nuevo intervalo que contiene a la raíz se renombran para el siguiente paso como a y c.

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

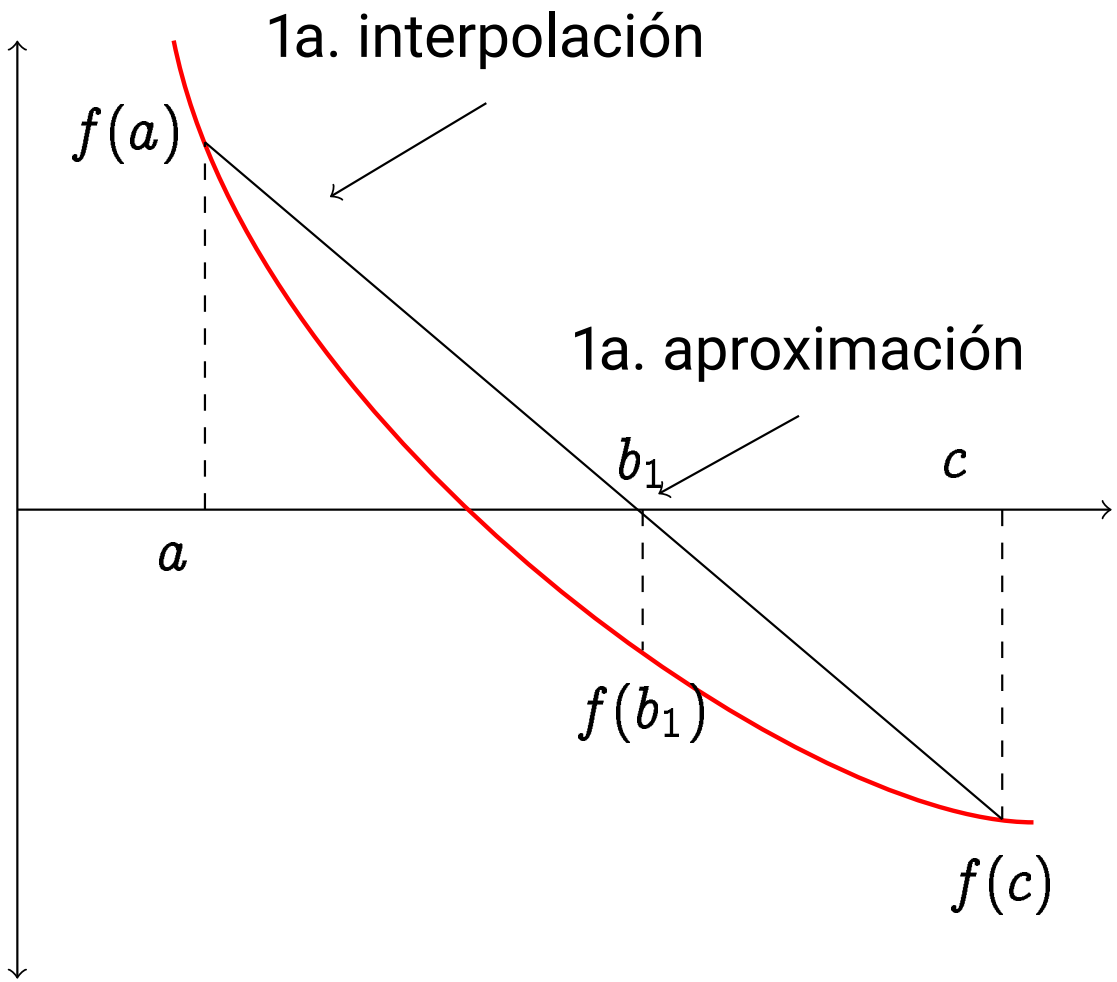
Método de la falsa posición



Método de la falsa posición



Método de la falsa posición



Método de la falsa posición

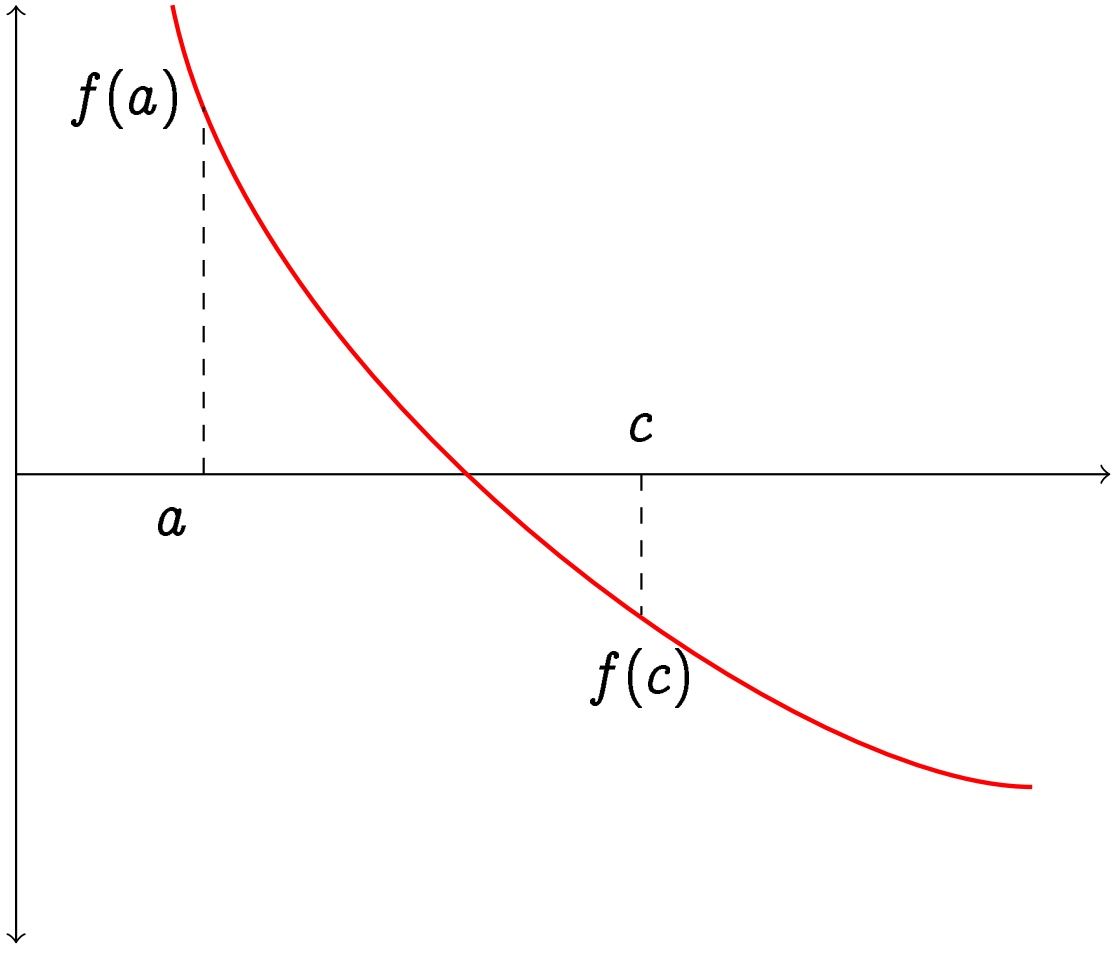


Figura 13: *Se renombran los extremos y se realiza la segunda aproximación lineal.*

Método de la falsa posición

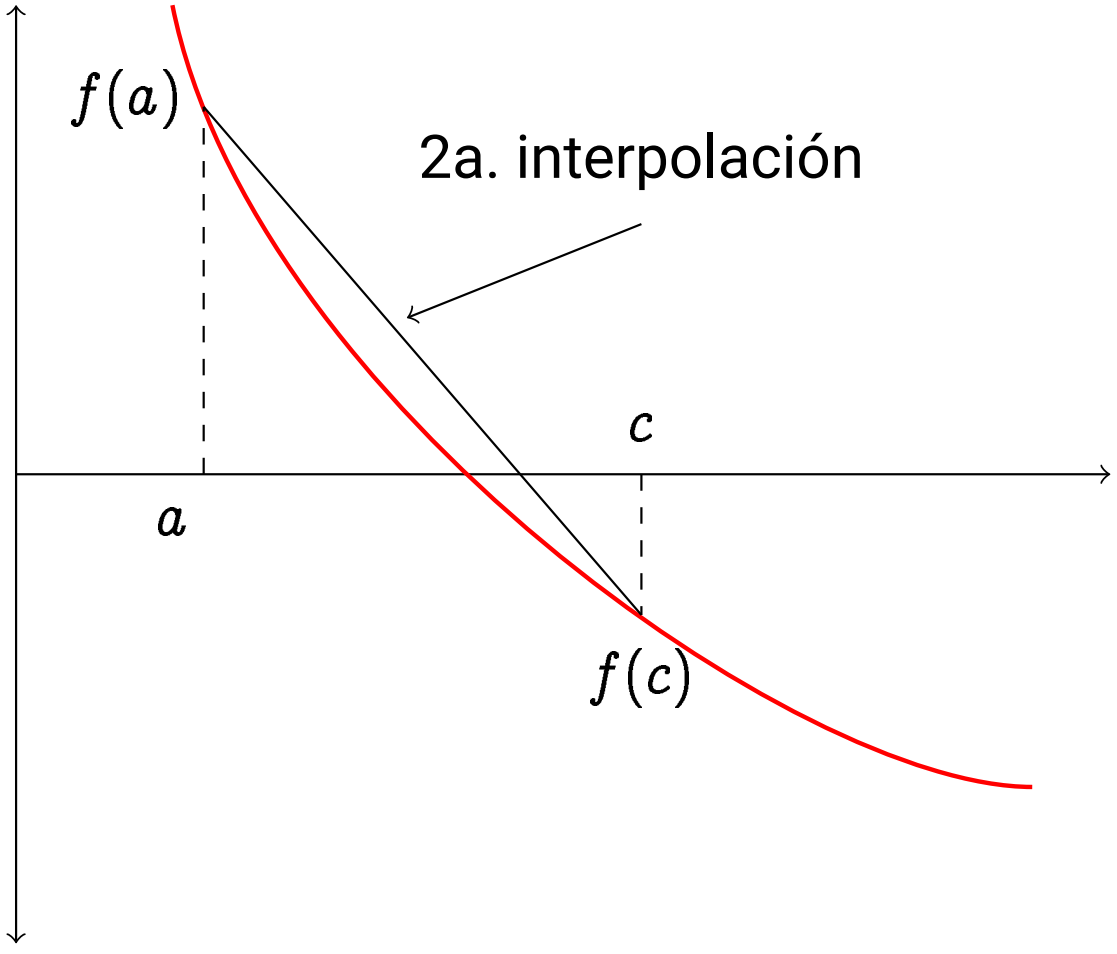


Figura 13: *Se renombran los extremos y se realiza la segunda aproximación lineal.*

Método de la falsa posición

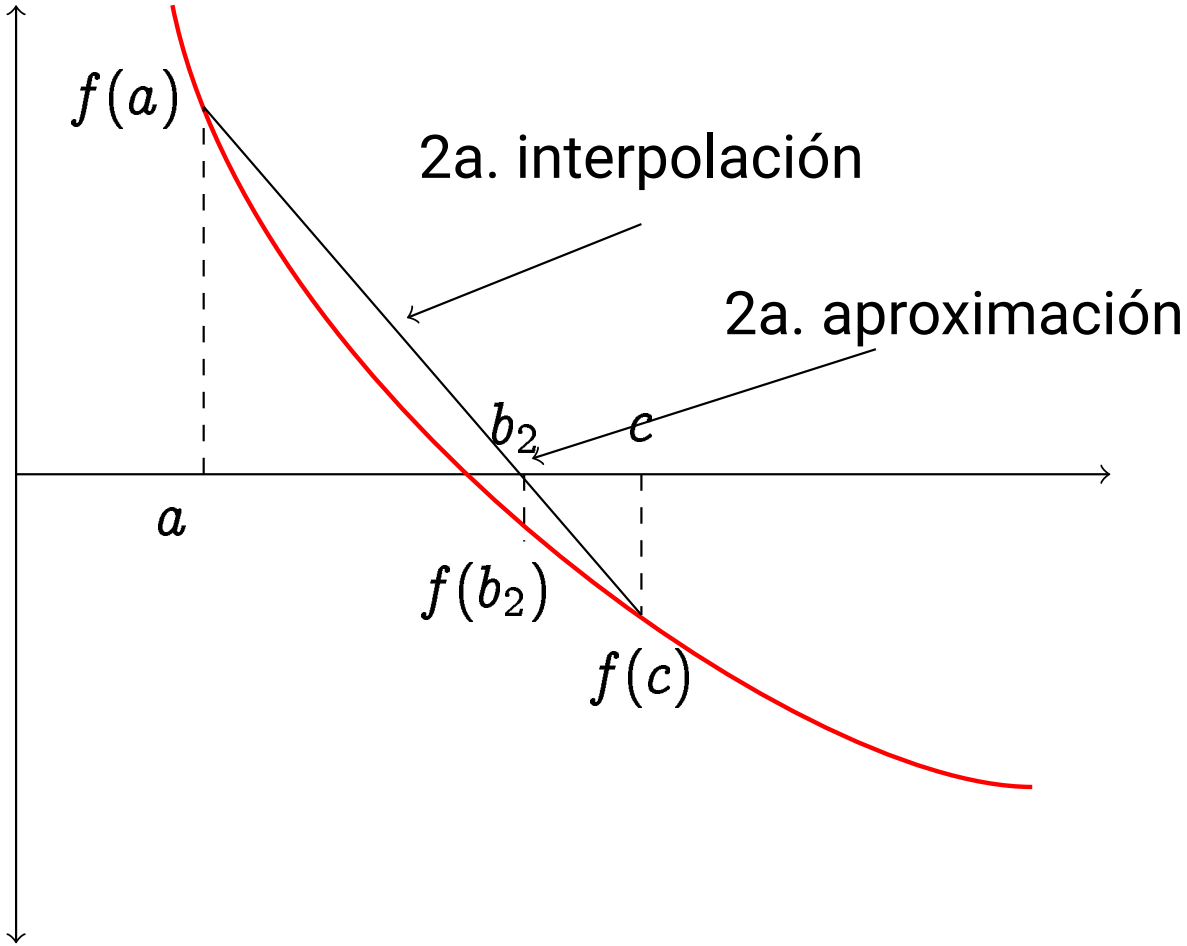


Figura 13: *Se renombran los extremos y se realiza la segunda aproximación lineal.*

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.

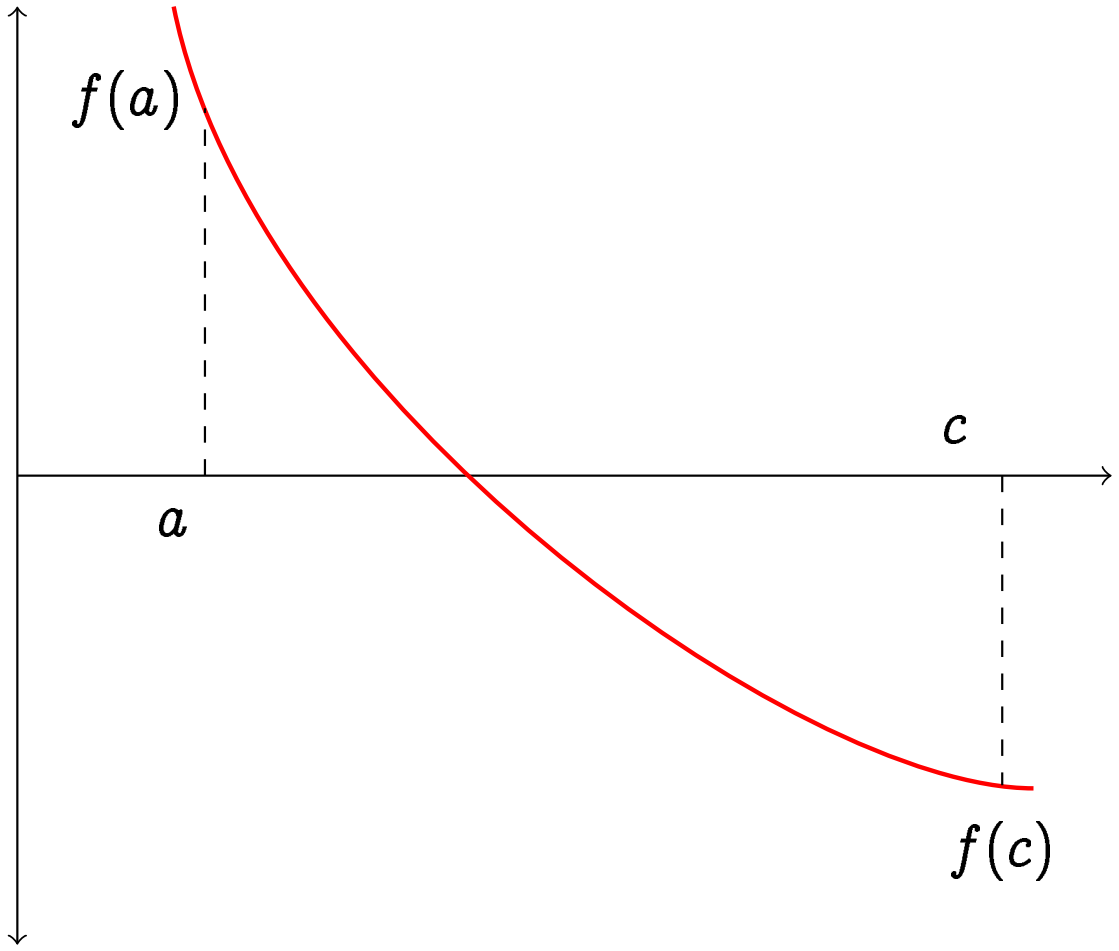
¿Qué podemos hacer al respecto?

Método de la falsa posición modificado

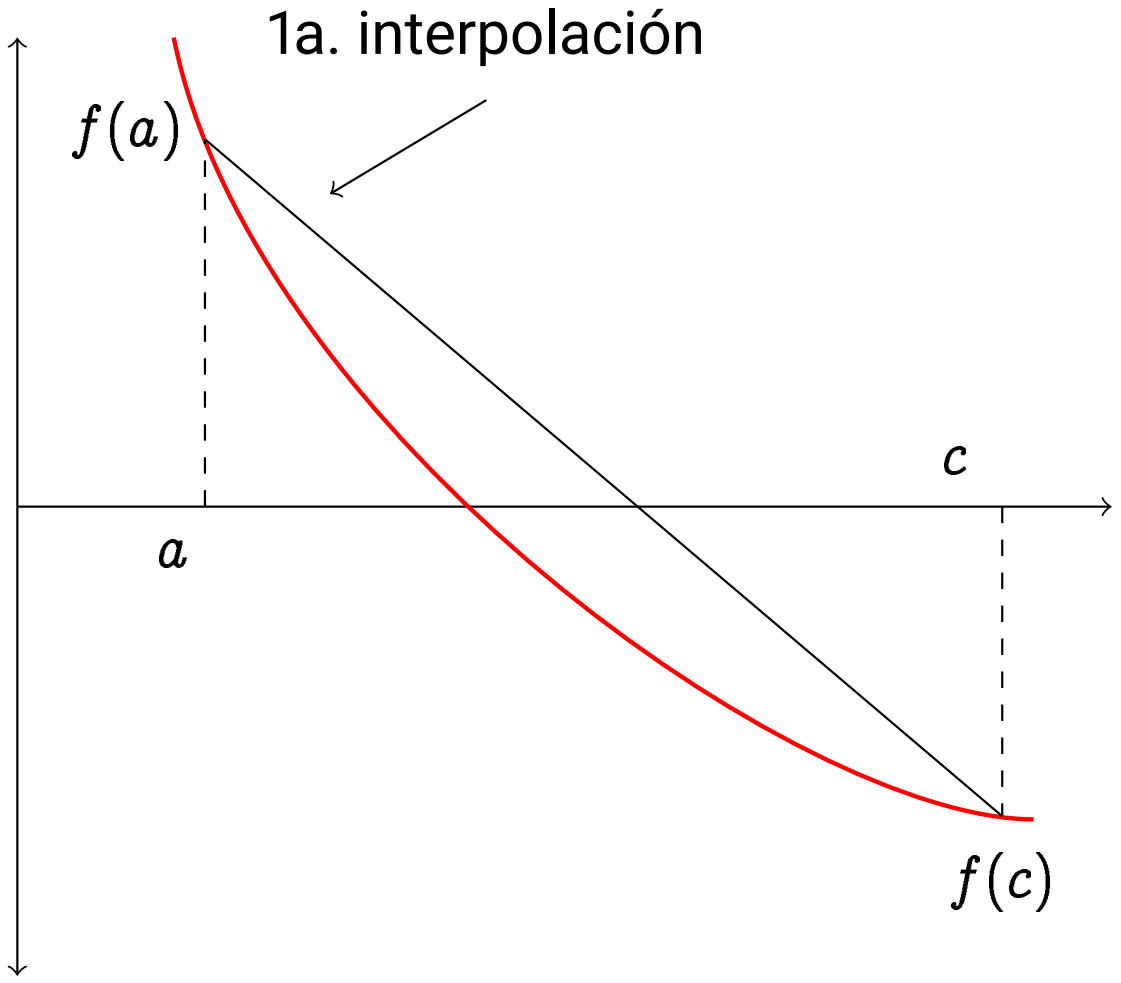
En este método, el valor de f en un punto fijo se divide a la mitad si este punto se ha repetido más de dos veces.

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.

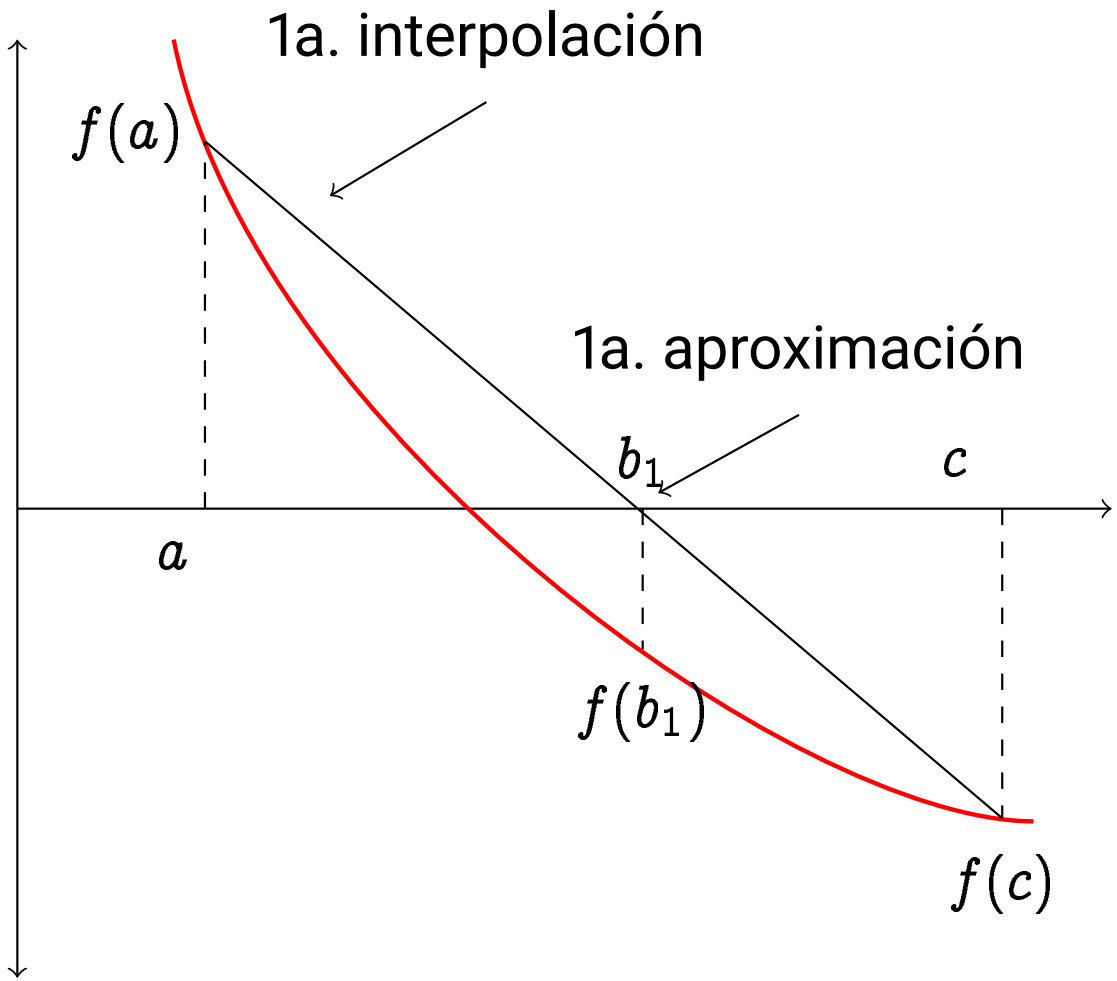
Método de falsa posición modificado



Método de falsa posición modificado



Método de falsa posición modificado



Método de la falsa posición modificado

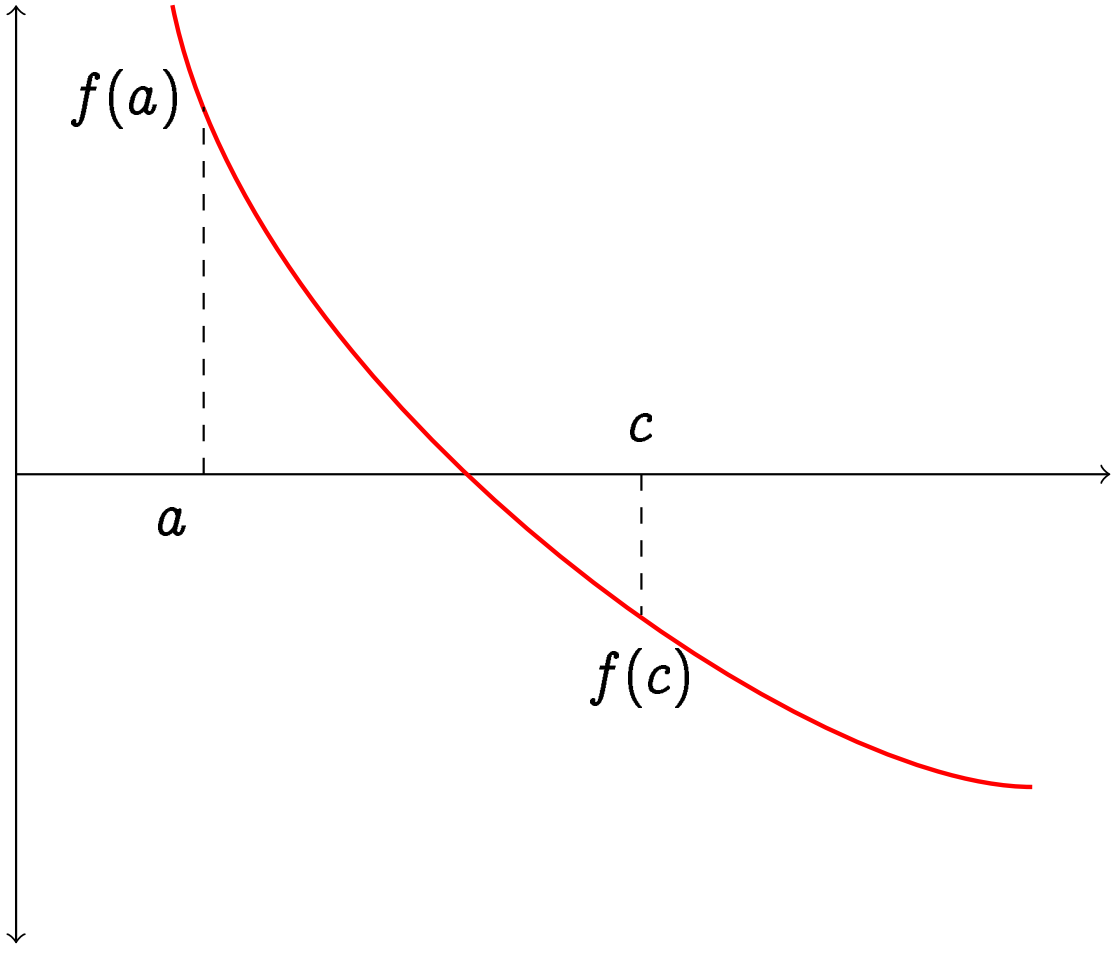


Figura 14: *Se evitan los extremos fijos de tal manera que se acelera la convergencia.*

Método de la falsa posición modificado

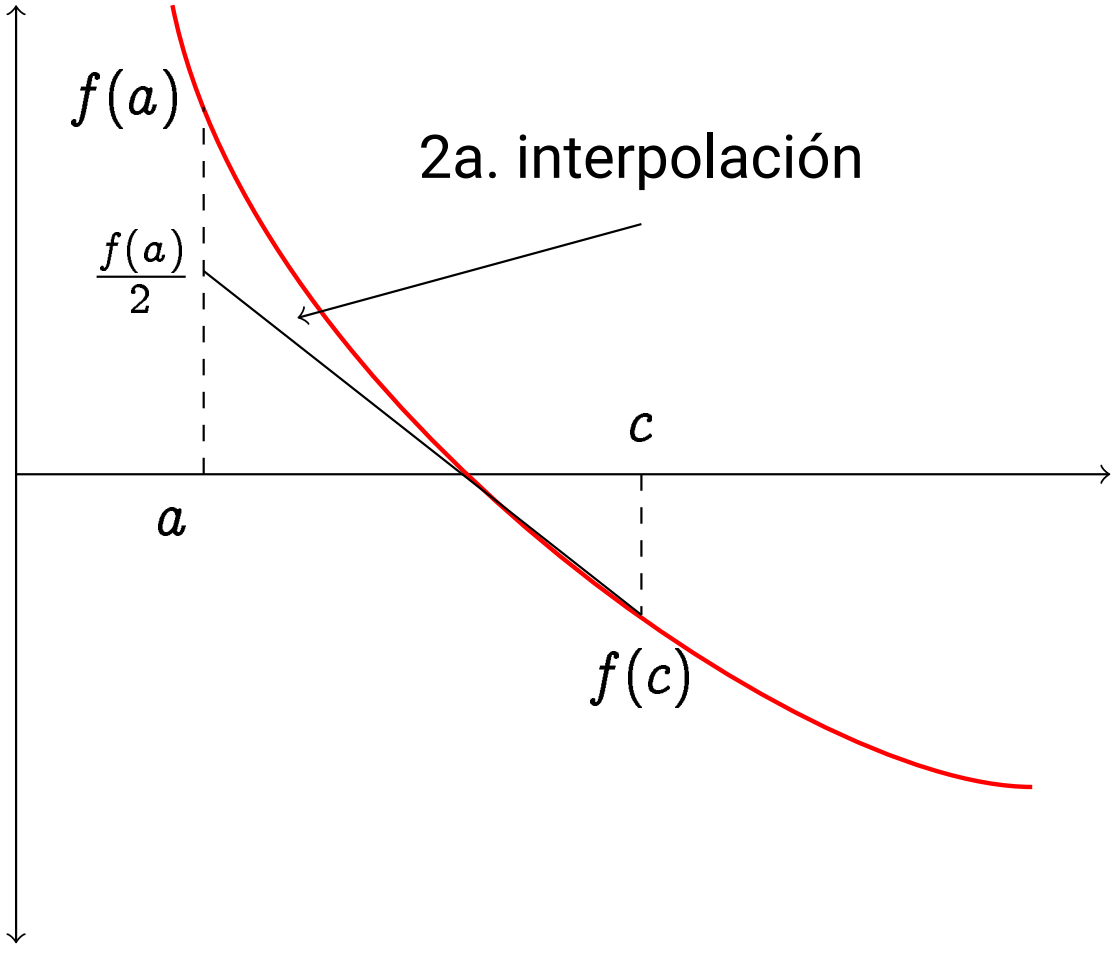


Figura 14: *Se evitan los extremos fijos de tal manera que se acelera la convergencia.*

Método de la falsa posición modificado

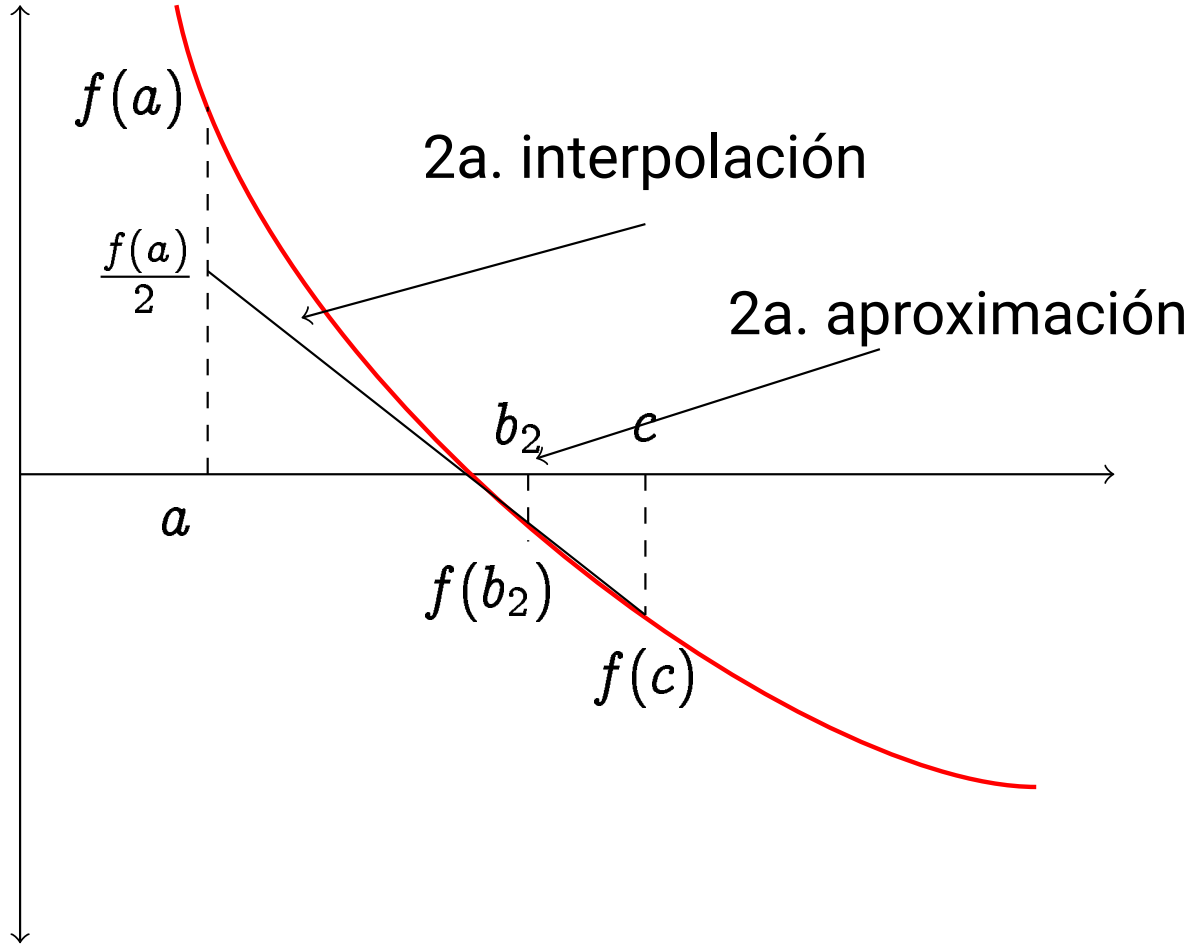


Figura 14: *Se evitan los extremos fijos de tal manera que se acelera la convergencia.*

Método de la secante

A diferencia del método de Newton, el valor de f' se aproxima utilizando dos valores de iteraciones consecutivas de f.

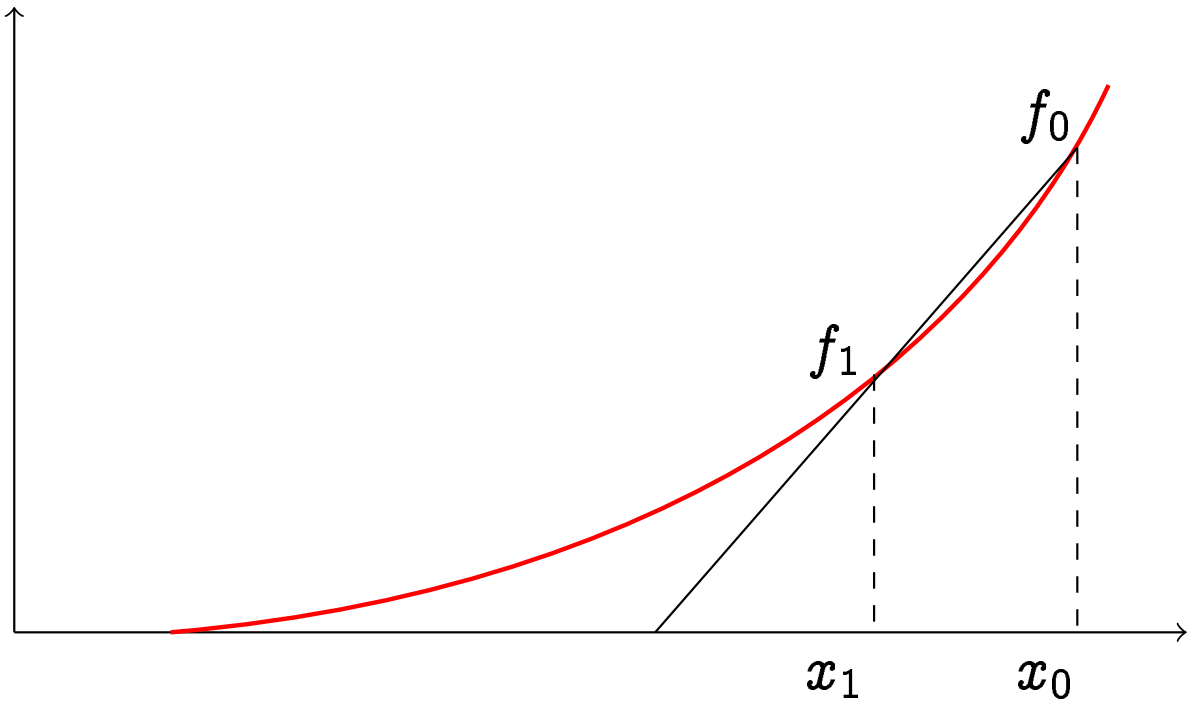
Con lo que se elimina la necesidad de evaluar tanto a f como a f' en cada iteración.

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

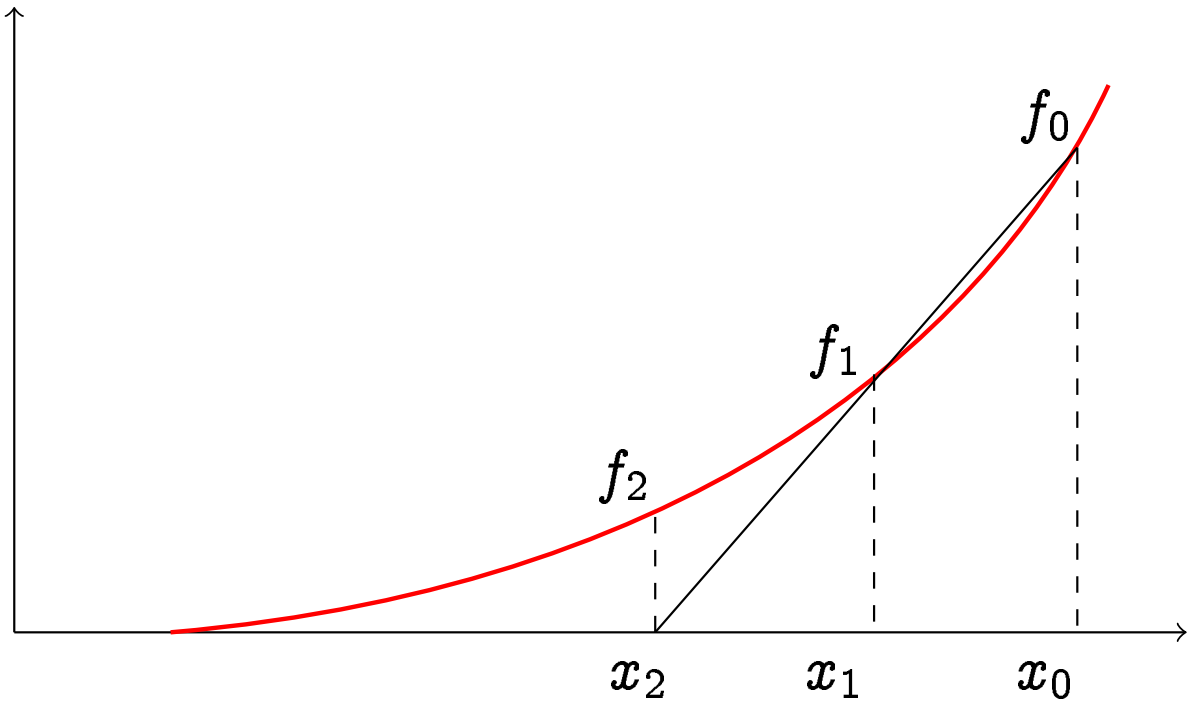
$$x_n = x_{n-1} - y_{n-1}\frac{x_{n-1} - x_{n-2}}{y_{n-1} - y_{n-2}}, \qquad n = 2, 3, \ldots$$

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

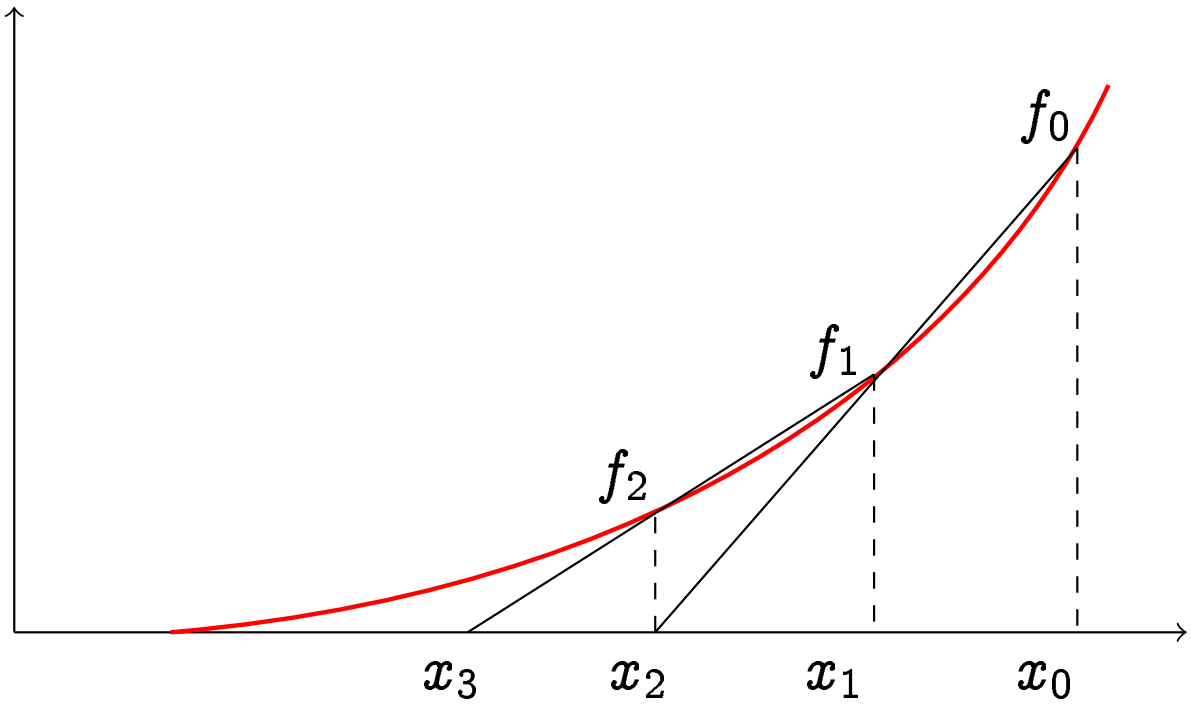
Método de la secante



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.