Grado Ingeniería Informática. Matemáticas 2. Prácticas.

Practica 5. Resolución de ecuaciones con métodos iterativos.

Departamento de Ciencia de la Computación e Inteligencia Artificial, Universidad de Alicante

28 de marzo de 2017

Introducción

Justificación

El problema es: dado $f(x)$ encontrar x_0 tal que $f(x_0) = 0$
Es interesante para

- *Resolver problemas de optimización:* Generalmente, se formulan tal que $f'(x) = 0$ (encontrar puntos críticos).
- *Factorización de integrales:* Dada una integral racional, descomponer el denominador en factores

$$\int \frac{a_0x^m + a_1x^{m-1} + \ldots + a_m}{b_0x^n + b_1x^{n-1} + \ldots + b_n} dx$$

.

- *Valores propios de matrices*: $q(\lambda) = |\mathbf{A} - \lambda\mathbf{I}|$ es la **ecuación característica**, las raíces reales son los valores propios reales λ de la matriz $\mathbf{A}$.

Introducción

Ejemplo

Encontrar un x_0 tal que $f(x_0) = 0$ para $f(x) = x^2 - sin(x) - 0.5$

$>> syms\ x;$

$>> f(x) = x^\wedge 2 - sin(x) - 0.5$

$>> ezplot(f)$

$>> hold\ on$

$>> line([-10\ 10],[0\ 0],'color','r')$

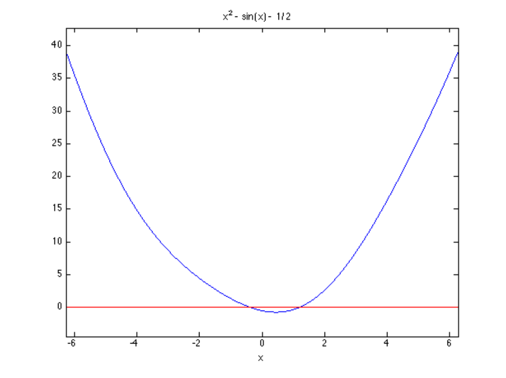
La recta da el intervalo $[a, b]$ donde $f(a)f(b) < 0$ de acuerdo con el Teorema de Bolzano:

Si $f(x)$ es continua en $[a, b]$ y $f(a)f(b) < 0$, entonces existe al menos un punto $c \in [a, b]$ tal que $f(c) = 0$

Introducción

Ejemplo

Encontrar un x_0 tal que $f(x_0) = 0$ para $f(x) = x^2 - sin(x) - 0.5$



Introducción

Ejemplo

Encontrar un x_0 tal que $f(x_0) = 0$ para $f(x) = x^2 - sin(x) - 0.5$
$[0, 2]$ es adecuado ya que contiene una de las dos raíces.

Si $a = 0$ y $b = 2$ comprobamos que $f(a)f(b) < 0$:
$>> a = 0; b = 2$
$>> sign(subs(f, a) * subs(f, b))$
$ans =$
-1

Veamos que sucede en el punto medio del intervalo:
$>> c = (a + b)/2;$

Introducción

Ejemplo

Encontrar un x_0 tal que $f(x_0) = 0$ para $f(x) = x^2 - sin(x) - 0.5$

- ¿Qué vale $f(c)$?
  $>> double(subs(f, c))$
  $ans =$
  -0.3415
- Entonces: $f(a) = -0.5$, $f(c) = -0.3415$ y $f(b) = 2.5907$.
- Luego la raíz está entre a y c puesto que $f(a)f(c) < 0$.
- Por lo tanto el **nuevo valor de** b debe ser c y el nuevo intervalo $[a, b = c]$.
- Este razonamiento conduce al Método de la Bisección.

Método de la Bisección

Definición

- Entrada: $f(x)$ y $[a, b]$.
- Encontrar $[a_i, b_i] \subset [a_{i+1}, b_{i+1}]$, con $f(a_i)f(b_i) < 0, \ \forall i$.
- Algoritmo:
  1. Sea $c_i = (a_i + b_i)/2$.
  2. Si $f(a_i)f(c_i) < 0$, entonces $b_{i+1} = c_i$, $a_{i+1} = a_i$.
     en caso contrario $a_{i+1} = c_i$, $b_{i+1} = b_i$.
  3. Paramos si
     - $h_i = |b_i - a_i|/2 < \Delta$
     - $|f(c_i)| < \epsilon$
     - $i =$ ciento valor dado.
  4. En caso de que no se cumpla ninguna de las condiciones de parada $i = i + 1$ y volver al punto 1.

Es un **método cerrado**, lo que significa que la raíz está en $[a_i, b_i]$.

Método de la Bisección

Para $[a, b] = [0, 2]$ se tiene $f(1.1963) = -0.0015836$

i	a	c	b	$(b-a)/2$
1	0	1	2	1
2	1	1.5	2	0.5
3	1	1.25	1.5	0.25
4	1	1.125	1.25	0.125
5	1.125	1.1875	1.25	0.0625
6	1.1875	1.2188	1.25	0.03125
7	1.1875	1.2031	1.2188	0.015625
8	1.1875	1.1953	1.2031	0.0078125
9	1.1953	1.1992	1.2031	0.0039062
10	1.1953	1.1973	1.1992	0.0019531
11	1.1953	1.1963	1.1973	0.00097656

Método de la Bisección

Para $[a, b] = [-1, 0]$ se tiene $f(-0.37012) = -0.0012839$

i	a	c	b	$(b-a)/2$
1	-1	-0.5	0	0.5
2	-0.5	-0.25	0	0.25
3	-0.5	-0.375	-0.25	0.125
4	-0.375	-0.3125	-0.25	0.0625
5	-0.375	-0.34375	-0.3125	0.03125
6	-0.375	-0.35938	-0.34375	0.015625
7	-0.375	-0.36719	-0.35938	0.0078125
8	-0.375	-0.37109	-0.36719	0.0039062
9	-0.37109	-0.36914	-0.36719	0.0019531
10	-0.37109	-0.37012	-0.36914	0.00097656

Método de la Secante

Definición

- Entrada: $f(x)$ y $[a, b]$.
- Encontrar $[a_i, b_i] \subset [a, b]$
- Algoritmo:
  1. Si $|f(a_i)| > |f(b_i)|$ entonces intercambiar a_i y b_i
  2. Sea $c_i = a_i - h_i$ donde $h_i = \frac{f(a_i)(b_i - a_i)}{f(b_i) - f(a_i)}$
  3. Sea $b_i = c_i$
  4. Paramos si
     - $h_i < \Delta$
     - $|f(c_i)| < \epsilon$
     - $i =$ ciento valor dado.
  5. En caso de que no se cumpla ninguna de las condiciones de parada $i = i + 1$ y volver al punto 1.

Es un **método abierto**, lo que significa que no se tiene seguridad de que la raíz está en $[a_i, b_i]$.

Método de la Secante.

Para $[a, b] = [0, 2]$ se tiene $f(-0.3709) = 2.1190 * 10^{-5}$

i	a	b	h	c	$f(c)$
1	0	2	-0.3236	0.3236	-0.7133
2	0	0.3236	0.7586	-0.7586	0.7634
3	0	-0.7586	0.3002	-0.3002	-0.1141
4	-0.3002	0	0.0888	-0.3890	0.0306
5	-0.3890	-0.3002	-0.0188	-0.3702	-0.0011
6	-0.3702	-0.3890	6.4446e-04	-0.3709	2.1190e-05

Método de la Secante.

Para $[a,b] = [-1,0]$ se tiene $f(-0.3709) = 2.1190 * 10^{-5}$

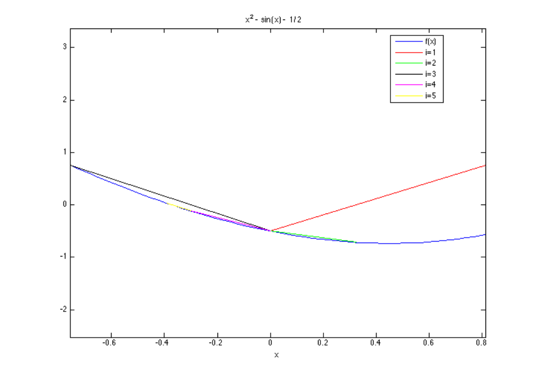
i	a	b	h	c	$f(c)$
1	0	-1	0.2715	-0.2715	-01581
2	-0.2715	0	0.1255	-0.3971	0.0444
3	-0.3971	-0.2715	-0.0275	-0.3695	-0.0022
4	-0.3695	-0.3971	0.0013	-0.3709	2.1190e-05

En ambos casos el Método Secante converge a la misma raíz, pero en **menos iteraciones** que en el Método de la Bisección.

Método de la Secante

Gráfica

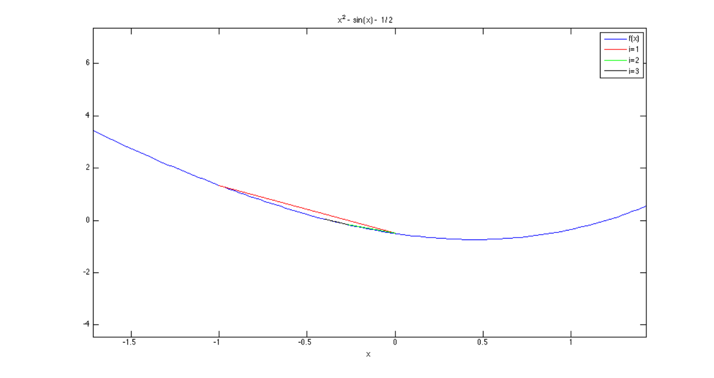
Resolución $f(x) - x^2 - sin(x) - 0.5 = 0$ en $[a, b] = [0, 2]$



Método de la Secante

Gráfica

Resolución $f(x) - x^2 - sin(x) - 0.5 = 0$ en $[a, b] = [-1, 0]$



Método de Regula Falsi

Definición

- Entrada: $f(x)$ y $[a, b]$.
- Encontrar $[a_i, b_i] \subset [a_{i+1}, b_{i+1}]$, con $f(a_i)f(b_i) < 0 \ \forall i$
- Algoritmo:
  1. Si $|f(a_i)| > |f(b_i)|$ entonces intercambiar a_i y b_i.
  2. Sea $c_i = a_i - h_i$ donde $h_i = \frac{f(a_i)(b_i - a_i)}{f(b_i) - f(a_i)}$
  3. Si $f(a_i)f(c_i) < 0$ entonces $b_{i+1} = c_i$, $a_{i+1} = a_i$;
     en caso contrario $a_{i+1} = c_i$, $b_{i+1} = b_i$;
  4. Paramos si
     - $h_i < \Delta$
     - $|f(c_i)| < \epsilon$
     - $i =$ ciento valor dado.
  5. En caso de que no se cumpla ninguna de las condiciones de parada $i = i + 1$ y volver al punto 1.

Es un método **cerrado**, lo que significa que la raíz está en $[a_i, b_i]$ ... Bisección.

Método de Regula Falsi

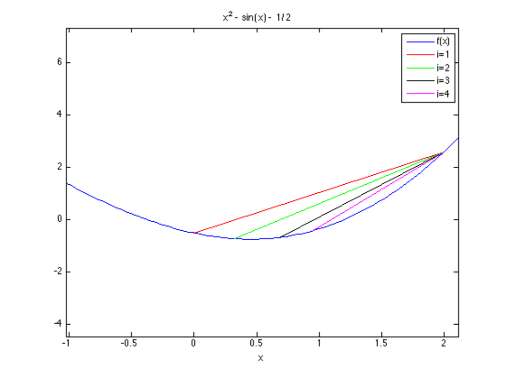
Para $[a, b] = [0, 2]$ tenemos $f(1.1958) = -5.7133 * 10^{-4}$

i	a	b	h	c	$f(c)$
1	0	**2**	-0.3236	0.3236	-0.7133
2	0.3236	**2**	-0.3619	0.6855	-0.6632
3	0.6855	**2**	-0.2679	0.9534	-0.4064
4	0.9534	**2**	-0.1419	1.0953	-0.1894
5	1.0953	**2**	-0.0616	1.1569	-0.0771
6	1.1569	**2**	-0.0244	1.1813	-0.0296
7	1.1813	**2**	-0.0093	1.1906	-0.0112
8	1.1906	**2**	-0.0035	1.1940	-0.0042
..	..	..	..	..	..
10	1.1953	**2**	-4.8592e-04	1.1958	-5.7133e-04

Método de Regula Falsi

Gráfica

Encontrar $f(x) - x^2 - sin(x) - 0.5 = 0$ en $[a, b] = [0, 2]$



Método de Regula Falsi

Para $[a, b] = [-1, 0]$ se tiene $f(-0.3707) = -3.1354 * 10^{-4}$

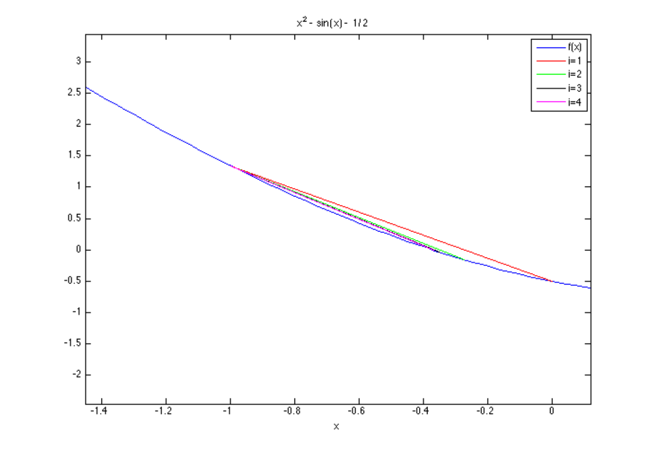
i	a	b	h	c	$f(c)$
1	0	**-1**	0.2715	-0.2715	-01581
2	-0.2715	**-1**	0.0768	-0.3483	-0.0374
3	-0.3483	**-1**	0.0177	-0.3660	-0.0082
4	-0.3660	**-1**	0.0039	-0.3698	-0.0018
5	-0.3698	**-1**	8.3164e-04	-0.3707	-3.1354e-04

El Método converge a la misma raíz, pero en **más iteraciones** que las obtenidas en el Método de la Secante.

Método de Regula Falsi

Gráfica

Resolución $f(x) - x^2 - sin(x) - 0.5 = 0$ en $[a, b] = [-1, 0]$



Series de Taylor

Aproximación de funciones

- El Método de la Secante se basa en aproximar $f(x)$ por una recta entre $(a, f(a))$ y $(b, f(b))$.
- Si $f(a)f(b) < 0$ la recta puede pasar por la raíz (Regula Falsi).
- Aproximar una función parece ser un buen método para encontrar raíces.
- Las rectas son la forma más sencilla de aproximar $f(x)$.

Las series de Taylor aproximan funciones mediante polinomios: Dado $x_0 \in [a, b]$, $f(x)$ se puede aproximar $f(x)$ en este intervalo mediante:

$$f(x) \approx f(x_0) + \frac{f^{'}(x_0)}{1!}(x - x_0) + \frac{f^{''}(x_0)}{2!}(x - x_0)^2 + \ldots + \frac{f^{(n)}(x_0)}{n!}(x - x_0)^n$$

Series de Taylor

Error de la aproximación

El error de aproximación con n derivadas viene dado por

$$R_n(x,\epsilon) = \frac{f^{(n+1)}(\epsilon)}{(n+1)!}(x-x_0)^{(n+1)}$$

para $\epsilon \in (x_0, x)$.
Es decir,

$$f(x) = P_n(x) + R_n(x,\epsilon)$$

donde

$$P_n(x) = f(x_0) + \frac{f'(x_0)}{1!}(x-x_0) + \frac{f''(x_0)}{2!}(x-x_0)^2 + \ldots + \frac{f^{(n)}(x_0)}{n!}(x-x_0)^n$$

Series de Taylor

Series de Taylor en MATLAB

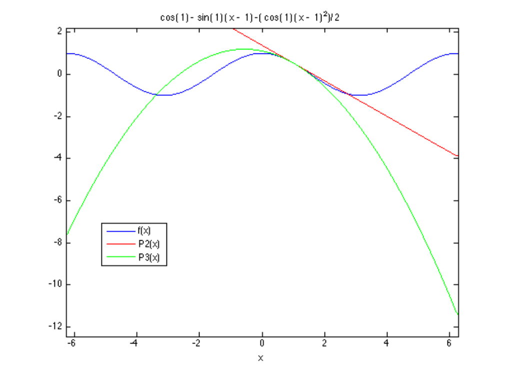
El comando *taylor*(f,v,a) calcula la Serie de Taylor de $n = 5$ (por defecto) de f con respecto a la variable v en el punto a:

$>> taylor(cos(x), x, 1,'Order', 1)$
$ans = cos(1)$
$>> taylor(cos(x), x, 1,'Order', 2)$
$ans = cos(1) - sin(1) * (x - 1)$
$>> taylor(cos(x), x, 1,'Order', 3)$
$ans = cos(1) - sin(1) * (x - 1) - (cos(1) * (x - 1)\hat{}2)/2$

Series de Taylor

Gráfica

Aproximación entorno al punto $a = 1$



Método de Newton

Segundo orden en la Serie de Taylor

1. Sea $a = x_0$
2. $f(x) \approx f(a) + f^{'}(a)(x - a)$
3. Sea $x = a - h$
4. por ejemplo, $f(x) \approx f(a) + f^{'}(a)(a - h - a) = f(a)hf^{'}(a)$
5. Si x es la raíz que estamos buscando, entonces

$$f(x) = 0 \text{ and } h = \frac{f(a)}{f'(a)}$$

Método de Newton

Definición

- Entrada: $f(x)$, a_1.
- Algoritmo:
  1. $h_i = \frac{f(a_i)}{f'(a_i)}$
  2. Sea $c_i = a_i - h_i$
  3. Paramos si
     - $h_i < \Delta$
     - $|f(c_i)| < \epsilon$
     - $i =$ ciento valor dado.
  4. En caso de que no se cumpla ninguna de las condiciones de parada $i = i + 1$, $a_i = c_i$ y volver al punto 1.

Es un método **abierto**, lo que significa que no es seguro que se alcance la raíz.

Método de Newton

Para $a_1 = 2$ tenemos $f(1.1966) = 0.0010$

i	a	h	c	$f(c)$
1	**2**	0.5866	1.4134	0.5099
2	**1.4134**	0.1910	1.2224	0.0543
3	**1.2224**	0.0258	1.1966	0.0010

En solo 3 iteraciones se puede aproximar la raíz y $f(c)$ disminuye un orden de magnitud por cada iteración!

Dibujar tangentes:

$>> ezplot(f)$

$>> hold\ on;$

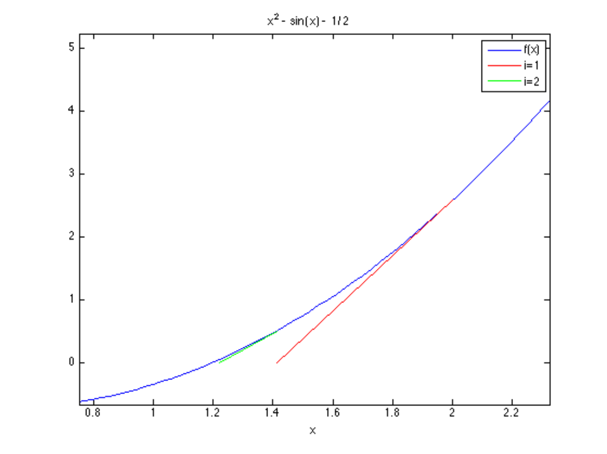
$>> line([XNewton\{1\}.a\ XNewton\{2\}.a], [XNewton\{1\}.fa\ 0], 'color', 'r')$

$>> line([XNewton\{2\}.a\ XNewton\{3\}.a], [XNewton\{2\}.fa\ 0], 'color', 'g')$

Método de Newton

Gráfica

Resolución $f(x) - x^2 - sin(x) - 0.5 = 0$ para $a_1 = 2$



Método de Newton

Para $a_1 = -1$ tenemos $f(-0.3709) = 1.5645 * 10^{-5}$

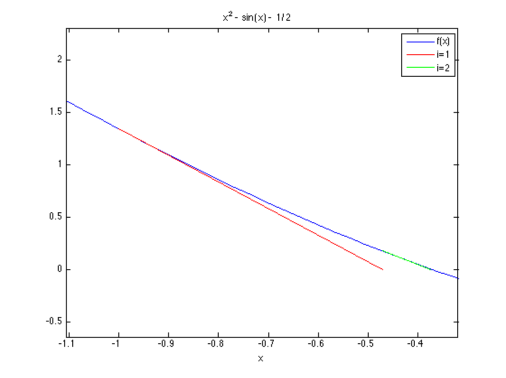
i	a	h	c	$f(c)$
1	**-1**	-0.5281	-0.4719	0.1773
2	**-0.4719**	-0.0967	-0.3753	0.0074
3	**-0.3709**	-0.0044	-0.3709	1.5645e-05

También en solo 3 iteraciones se alcanza una buena aproximación de la raíz. Compara con otros métodos!

Método de Newton

Gráfica

Resolución $f(x) - x^2 - sin(x) - 0.5 = 0$ para $a_1 = -1$



Ejercicios

Práctica #1

Construir una función llamada Bisection(f,a,b,tolerancia,errorfun,maxiter) que implemente el Método de la Bisección, teniendo en cuenta que el error en la raíz es la *tolerancia* $= \Delta$, el error en la función es *errorfun* $= \epsilon$ y el número máximo de iteraciones es *maxiter*.

- Probarlo con la función $f(x) = f(x) = x^2 - sin(x) - 0.5$
- Implementar la iteración con: while...end
- Obtener el valor de la raíz c para
  - Cuando $\Delta \leq 10^{-3}$
  - Cuando $\epsilon \leq 10^{-3}$
  - Cuando se realicen 4 iteraciones
- Probar los intervalos $[0, 2]$ y $[-1, 0]$
- ¿Cuantas iteraciones se necesitan si $\Delta \leq 10^{-5}$?

Ejercicios

Práctica #1 (cont)

Construir una estructura llamada X que almacene la tabla que almacena los resultados intermedios. Esta estructura se devuelve junto con r.

- Al principio $i = 1$.
- Para el primer intervalo $X\{i\}.a = a1; X\{i\}.b = b1;$.
- Al calcular $c = (a1 + b1)/2$ realizar $X\{i\}.c = c;$
  $X\{i\}.fa = double(subs(f, a));$
  $X\{i\}.fc = double(subs(f, c));$
  $X\{i\}.fb = double(subs(f, b));$
- Después de decidir el nuevo intervalo, incrementar i y almacenarlo en $X\{i\}.a = a1; X\{i\}.b = b1;$

Ejercicios

Práctica #2

Construir un programa llamado Secant(f,a,b,tolerancia,errorfun,maxiter) que implemente el Método de la Secante, teniendo en cuenta que el error en la raíz es la $tolerancia = \Delta$, el error en la función es $errorfun = \epsilon$ y el número máximo de iteraciones es *maxiter*.

- Probarlo con la función $f(x) = f(x) = x^2 - sin(x) - 0{,}5$
- Implementar la iteración con: while... end
- Obtener el valor de la raíz c
  - Cuando $\Delta \leq 10^{-3}$
  - Cuando $\epsilon \leq 10^{-3}$
  - Cuando se realicen 4 iteraciones
- Probar los intervalos $[0, 2]$ y $[-1, 0]$
- ¿Cuantas iteraciones se necesitan si $\Delta \leq 10^{-5}$?

Ejercicios

Práctica #2 (cont)

Construir una estructura X que almacene la tabla como en la práctica #1 y usarla para mostrar las gráficas de las secantes. De forma que, si $Xsec$ es la estructura para el intervalo $[0, 2]$:

```
>> ezplot(f); hold on;
>> line([Xsec{1}.a Xsec{1}.b], [Xsec{1}.fa Xsec{1}.fb],'color','r')
>> line([Xsec{2}.a Xsec{2}.b], [Xsec{2}.fa Xsec{2}.fb],'color','g')
>> line([Xsec{3}.a Xsec{3}.b], [Xsec{3}.fa Xsec{3}.fb],'color','black')
>> line([Xsec{4}.a Xsec{4}.b], [Xsec{4}.fa Xsec{4}.fb],'color','magenta')
>> line([Xsec{5}.a Xsec{5}.b], [Xsec{5}.fa Xsec{5}.fb],'color','yellow')
```

Ejercicios

Práctica #3

Construir una función llamada RegulaFalsi(f,a,b,tolerancia,errorfun,maxiter) que implemente el Método de Regula Falsi, teniendo en cuenta que el error en la raíz es la $tolerancia = \Delta$, el error en la función es $errorfun = \epsilon$ y el número máximo de iteraciones es *maxiter*.

- Probarlo con la función $f(x) = f(x) = x^2 - sin(x) - 0{,}5$
- Implementar la iteración con: while...end
- Tiene que devolver c
    - Cuando $\Delta \leq 10^{-3}$
    - Cuando $\epsilon \leq 10^{-3}$
    - Cuando se realicen 4 iteraciones
- Probar dos intervalos $[0, 2]$ y $[-1, 0]$
- ¿Cuántas iteraciones necesita si $\Delta \leq 10^{-5}$?

Ejercicios

Práctica #4

Construir una función llamada Newton(f,a,tolerancia,errorfun,maxiter) que implemente el Método de Newton, teniendo en cuenta que el error en la raíz es la $tolerancia = \Delta$, el error en la función es $errorfun = \epsilon$ y el número máximo de iteraciones es *maxiter*.

- Probarla con $f(x) = f(x) = x^2 - sin(x) - 0{,}5$
- Implementar el bucle con: while...end
- Devolver c
  - Cuando $\Delta \leq 10^{-3}$
  - Cuando $\epsilon \leq 10^{-3}$
  - Cuando se realicen 4 iteraciones
- Probar para $a_1 = 2$ y $a_1 = -1$
- ¿Cuántas iteraciones se necesita para $\epsilon = 10^{-5}$?

Ejercicios

Práctica #5

Considerar la función $f(x) = \exp^x - 4x^2$, cuyas raíces están en los intervalos $[-1, 0], [0, 1]$ y $[4, 4.5]$. Encontrar la primera raíz usando el método de la Bisección, la segunda con el método de Regula Falsi y la tercera con el método de Newton

Ejercicios

Práctica #6

Construir una función llamada Steffensen(f,a,tolerancia,errorfun,maxiter) para implementar el Método de Steffensen.

En este método las iteraciones siguen la fórmula:

$$x_{k+1} = x_k - \frac{(f(x_k))^2}{f(x_k + f(x_k)) - f(x_k)}, \ k \geq 0$$