# Práctica 4. Resolución de ecuaciones

Matemáticas 2. Ingeniería Informática

# Introducción

## Justificación

Dada una función $f(x)$ el problema es **encontrar un valor** $x_0$ 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

$$\int \frac{a_0 x^m + a_1 x^{m-1} + \ldots + a_m}{b_0 x^n + b_1 x^{n-1} + \ldots + b_n} dx$$

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

# Introducción

## Ejemplo

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

$>> syms\ x;$

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

$>> ezplot(f)$

$>> hold\ on$

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

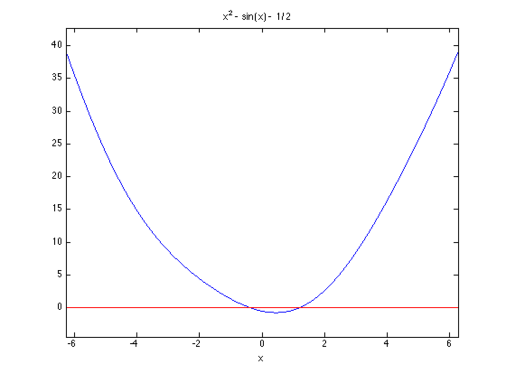
Según el **Teorema de Bolzano**

*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 $f(x) = x^2 - sin(x) - 0.5$
El intervalo $[0, 2]$ 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 $c$ y $b$ puesto que $f(b)f(c) < 0$
- Por lo tanto $c$ sustituye a $a$ y el **nuevo intervalo** es $[a = c, b]$
- Este razonamiento conduce al Método de la Bisección

# Bisección

## Definición

- Entrada: Una función, un intervalo, el error en la función, la tolerancia y el número de iteraciones: $f(x), a, b, \epsilon, \Delta, n$ con $f(a_i)f(b_i) < 0, \ \forall i$
- Salida: $c_i$
- Algoritmo
  1. Calculamos $c_i = (a_i + b_i)/2$, el punto medio del intervalo
  2. Comprobamos si cumple Bolzano, $f(a_i)f(c_i) < 0$, 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 $h_i < \Delta$ ó $|f(c_i)| < \epsilon$ ó $i >= n$
  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 $[a_i, b_i]$

# Bisección

## Pseudo-código del Método de la Bisección

```
BúsquedaPorBisección (f(x), a, b, ε, Δ, n)
    i:=0
    h:=abs(b-a)
    repetir
        i:=i+1
        c:=(a+b)/2
        h:=h/2
        si signo(f(a))*signo(f(c))<0
            entonces
                b:=c
            si no
                a:=c
    hasta (abs(f(c)) ≤ ε) ó (h ≤ Δ) ó (i = n)
devolver c
```

# Bisección

Para $f(x) = x^2 - sin(x) - 0.5$ en $[0, 2]$ 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 | 1.21875 | 1.25 | 0.03125 |
| 7 | 1.1875 | 1.203125 | 1.21875 | 0.015625 |
| 8 | 1.1875 | 1.195312 | 1.203125 | 0.0078125 |
| 9 | 1.195312 | 1.199219 | 1.203125 | 0.0039062 |
| 10 | 1.195312 | 1.197266 | 1.199219 | 0.0019531 |
| 11 | 1.195312 | 1.196289 | 1.197266 | 0.00097656 |

# Bisección

**Para $f(x) = x^2 - sin(x) - 0.5$ en $[-1, 0]$ se tiene $f(-0.37012) = -0.00128$**

| 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 | -0.359375 | -0.34375 | 0.015625 |
| 7 | -0.375 | -0.367188 | -0.359375 | 0.0078125 |
| 8 | -0.375 | -0.371094 | -0.367188 | 0.0039062 |
| 9 | -0.371094 | -0.369141 | -0.367188 | 0.0019531 |
| 10 | -0.371094 | -0.370117 | -0.369141 | 0.00097656 |

# Secante

## Definición

- Entrada: Una función, dos números, el error en la función, la tolerancia y el número de iteraciones: $f(x), a, b, \epsilon, \Delta, n$
- Salida: $c_i$
- Algoritmo
  1. Si $|f(a_i)| > |f(b_i)|$ entonces intercambiar $a_i$ y $b_i$
  2. Hacer $c_i = a_i - h_i$ donde $h_i = \frac{f(a_i)(b_i - a_i)}{f(b_i) - f(a_i)}$
  3. El nuevo valor de $c_i$ sustituye a $b_i$, es decir, $b_i = c_i$
  4. Paramos si $h_i < \Delta$ ó $|f(c_i)| < \epsilon$ ó $i >= n$
  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 $[a_i, b_i]$

# Secante

## Pseudo-código del Método de la Secante

```
BúsquedaPorSecante (f(x), a, b, ε, Δ, n)
    i:=0
    repetir
        i:=i+1
        si abs(f(a))>abs(f(b)) entonces
            (*/ Intercambiar 'a' por 'b' /*)
            a⇔b
        h:=f(a)*(b-a)/(f(b)-f(a))
        c:=a-h
        b:=c
    hasta (abs(f(c))≤ε) ó (abs(h)≤Δ) ó (i=n)
devolver c
```

# Secante

**Para $f(x) = x^2 - sin(x) - 0.5$ en $[0, 2]$ se tiene $f(-0.3709) = 2.119 * 10^{-5}$**

A pesar de empezar con $a = 0$ y $b = 2$ se llega a la raíz entre $[-1, 0]$

| i | a | b | h | c | $f(c)$ |
|---|---|---|---|---|---|
| 1 | 0 | 2 | -0.323551 | 0.323551 | -0.713250 |
| 2 | 0 | 0.323551 | 0.758619 | -0.758619 | 0.763422 |
| 3 | 0 | -0.758619 | 0.300224 | -0.300224 | -0.114132 |
| 4 | -0.300224 | 0 | 0.0888 | -0.389024 | 0.030625 |
| 5 | -0.389024 | -0.300224 | -0.018786 | -0.370237 | -0.001088 |
| 6 | -0.370237 | -0.389024 | 0.000644 | -0.370882 | -0.000010 |

# Secante

**Para $f(x) = x^2 - sin(x) - 0.5$ en $[-1, 0]$ se tiene $f(-0.3709) = 2.1 * 10^{-5}$**

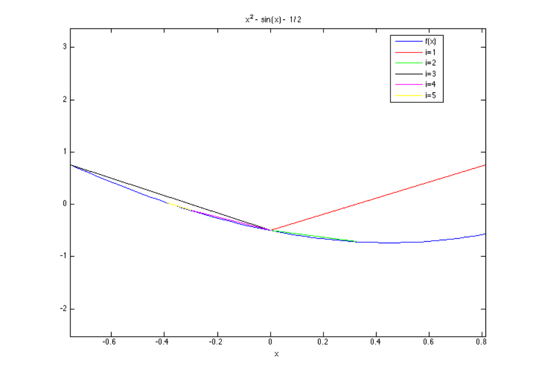
| i | a | b | h | c | $f(c)$ |
|---|---|---|---|---|---|
| 1 | 0 | -1 | 0.271522 | -0.271522 | -0158078 |
| 2 | -0.271522 | 0 | 0.125530 | -0.397052 | 0.044352 |
| 3 | -0.397052 | -0.271522 | -0.027504 | -0.369549 | -0.002239 |
| 4 | -0.369549 | -0.397052 | 0.001322 | -0.370870 | -0.000028 |

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

# Secante

## Gráfica

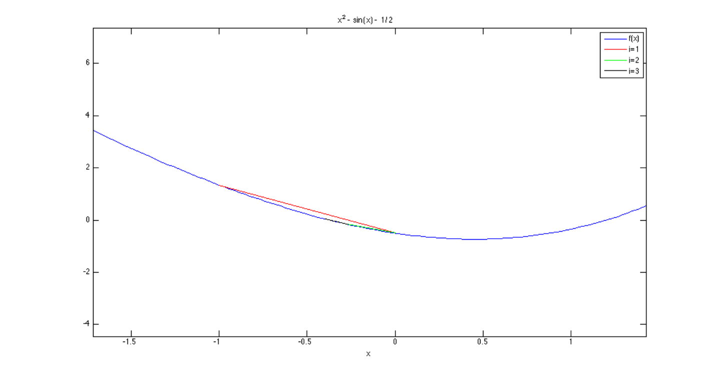
Resolución $f(x) - x^2 - sin(x) - 0.5 = 0$ en $[a, b] = [0, 2]$



# Secante

## Gráfica

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



# Regula Falsi

## Definición

- Entrada: Una función, un intervalo, el error en la función, la tolerancia y el número de iteraciones: $f(x), a, b, \epsilon, \Delta, n$ con $f(a_i)f(b_i) < 0, \ \forall i$
- Salida: $c_i$
- 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. Comprobamos si cumple Bolzano, $f(a_i)f(c_i) < 0$ entonces $b_{i+1} = c_i$ y $a_{i+1} = a_i$, en caso contrario $a_{i+1} = c_i$, $b_{i+1} = b_i$;
  4. Paramos si $h_i < \Delta$ ó $|f(c_i)| < \epsilon$ ó $i >= n$
  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 $[a_i, b_i]$

# Regula Falsi

## Pseudo-código del Método de la Regula Falsi

```
BúsquedaRegulaFalsi (f(x),a,b,ε,Δ,n)
    i:=0
    repetir
        i:=i+1
        si abs(f(a))>abs(f(b)) entonces
            a⇔b
        h:=f(a)*(b-a)/(f(b)-f(a))
        c:=a-h
        si signo(f(a))*signo(f(c))<0
            entonces b:=c
            si no      a:=c
    hasta (abs(f(c))≤ε) ó (abs(h)≤Δ) ó (i=n)
devolver c
```

# Regula Falsi

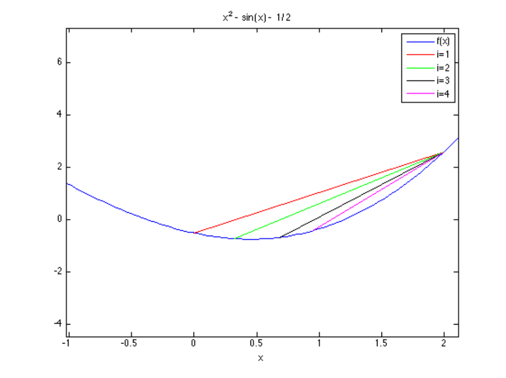
Para $f(x) = x^2 - sin(x) - 0.5$ en $[0, 2]$, $f(1.1958) = -5.7133 * 10^{-4}$

| i | a | b | h | c | $f(c)$ |
|---|---|---|---|---|---|
| 1 | 0 | **2** | -0.323551 | 0.323551 | -0.713250 |
| 2 | 0.323551 | **2** | -0.361908 | 0.685459 | -0.663174 |
| 3 | 0.685459 | **2** | -0.267917 | 0.953376 | -0.406448 |
| 4 | 0.953376 | **2** | -0.141934 | 1.095311 | -0.189365 |
| 5 | 1.095311 | **2** | -0.061623 | 1.156934 | -0.077078 |
| 6 | 1.156934 | **2** | -0.024358 | 1.181292 | -0.029647 |
| 7 | 1.181292 | **2** | -0.009263 | 1.190555 | -0.011154 |
| .. | .. | .. | .. | .. | .. |
| 10 | 1.195318 | **2** | -0.000481 | 1.195798 | -0.000575 |

# Regula Falsi

## Gráfica

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



# Regula Falsi

Para $f(x) = x^2 - sin(x) - 0.5$ en $[-1, 0]$, $f(-0.3707) = -3.1354 * 10^{-4}$

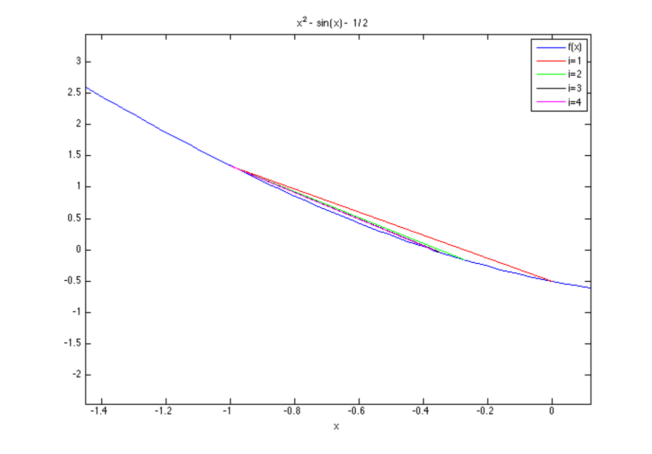
| i | a | b | h | c | $f(c)$ |
|---|---|---|---|---|---|
| 1 | 0 | **-1** | 0.271522 | -0.271522 | -0.158078 |
| 2 | -0.271522 | **-1** | 0.076794 | -0.348316 | -0.037361 |
| 3 | -0.348316 | **-1** | 0.017658 | -0.365974 | -0.008204 |
| 4 | -0.365974 | **-1** | 0.003854 | -0.369828 | -0.001772 |
| 5 | -0.369828 | **-1** | 0.000831 | -0.370659 | -0.000381 |

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

# Regula Falsi

## Gráfica

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



# Método de Newton

## Definición

- Entrada: Una función, un valor inicial, el error en la función, la tolerancia y el número de iteraciones: $f(x), a, \epsilon, \Delta, n$
- Salida: $c_i$
- Algoritmo:
  1. $h_i = \frac{f(a_i)}{f'(a_i)}$
  2. Sea $c_i = a_i - h_i$
  3. Paramos si $h_i < \Delta$ ó $|f(c_i)| < \epsilon$ ó $i >= n$
  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 **abierto**, no es seguro que se alcance la raíz!

# Newton

## Pseudo-código del Método de la Newton

```
BúsquedaPorNewton (f(x),a,ε,Δ,n)
    f'(x)::=df(x)/dx
    i:=0
    repetir
        i:=i+1
        h:=f(a)/f'(a)
        c:=a-h
        a:=c
    hasta (abs(f(c))≤ε) ó (abs(h)≤Δ) ó (i=n)
devolver c
```

# Método de Newton

**Para $a_1 = 2$ tenemos $f(1.196082) = 0.0$**

| i | a | h | c | $f(c)$ |
|---|---|---|---|---|
| 1 | **2** | 0.586643 | 1.413357 | 0.509945 |
| 2 | **1.413357** | 0.190996 | 1.222361 | 0.054257 |
| 3 | **1.222361** | 0.025796 | 1.196564 | 0.000977 |
| 4 | **1.196564** | 0.000482 | 1.196082 | 0.000000 |

En solo 3 iteraciones se puede aproximar la raíz.

Se pueden dibujar las rectas tangentes mediante el siguiente código Matlab

$>> ezplot(f), hold\ on;$

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

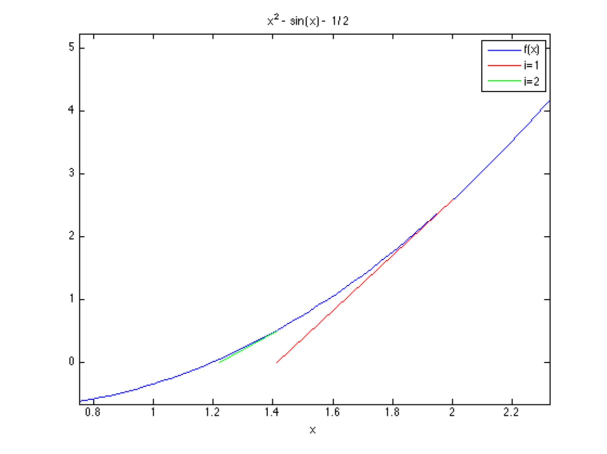
$>> line([XNewton(2).a\ XNewton(3).a], [XNewton(2).fa\ 0], 'color', 'g')$

*XNewton* es una estructura con los elementos necesarios en cada iteración.

# 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 $f(-0.370887) = 1.0$

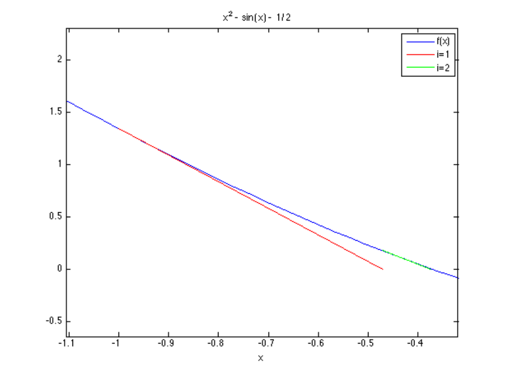
| i | a | h | c | $f(c)$ |
|---|---|---|---|---|
| 1 | **-1** | -0.528075 | -0.471925 | 0.177314 |
| 2 | -**0.471925** | -0.096653 | -0.375272 | 0.007354 |
| 3 | -**0.375272** | -0.004375 | -0.370897 | 0.000016 |
| 4 | -**0.370897** | -0.000009 | -0.370887 | 0.000000 |

En solo 3 iteraciones se alcanza una buena aproximación

# Método de Newton

## Gráfica

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



# Comando *fzero*

## Definición

Sirve para calcular raices de funciones. Se invoca de la siguiente manera *fzero(fun, inicial, op)* Donde *fun* es la función de la que se quiere calcular las raices, *inicial* es el valor de partida inicial -puede ser un intervalo- y *op* son parámetros de la función.

# Comando *fzero*

**Ejemplo 1**

$>> f = @(x)\ 0.4 - x + sin(x)$

$>> fzero(f, 0)$

# Comando *fzero*

## Ejemplo 2

$>> f = @(x)\ sqrt(9.81 * x/0.25) * tanh(sqrt(9.81 * 0.25/x) * 4) - 35', 80)$

$>> options = optimset('display',' iter')$

$>> fzero(f, 80, options)$

# Comando *fzero*

## Ejemplo 3

$>> fzero('cos(x) * cosh(x) + 1', [0, 5])$

$>> fzero('cos(x) * cosh(x) + 1', [3, 7])$

$>> fzero('cos(x) * cosh(x) + 1', [7, 10])$

$>> fzero('cos(x) * cosh(x) + 1', [0, 1])$

# Ejercicios

## Ejercicio #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

## Ejercicio #1 (cont)

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

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

# Ejercicios

## Ejercicio #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

## Ejercicio #2 (cont)

**Construir una estructura** $Xsec$ **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

## Ejercicio #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

## Ejercicio #3 (cont)

**Construir una estructura** *Xreg* **que almacene la tabla y usarla para mostrar las gráficas de las secantes. De forma que, si** *Xreg* **es la estructura para el intervalo** $[0, 2]$

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

# Ejercicios

## Ejercicio #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

## Ejercicio #4 (cont)

**Construir una estructura *Xnew* que almacene la tabla y usarla para mostrar las gráficas de las tangentes.**

# Ejercicios

## Ejercicio #5

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

# Ejercicios

## Ejercicio #6

**Encontrar graficamente una solución de la ecuación:** $xe^x - 7 = 0$

# Ejercicios

## Ejercicio #7

**Utilizando la función fzero calcula**

- Una solución de la ecuación $xe^x - 7 = 0$, con $x_0 = 1$.
- Una solución de la ecuación $xe^x - 7 = 0$, en el intervalo $[1, 2]$.

# Ejercicios

## Ejercicio #8

**Crea una función llamada CalculoRaices, que tenga como parametros de entrada una función matemática y un intervalo, y que devuelva las raices de la función (utiliza fzero).**

# Ejercicios

## Ejercicio #9

**Construir una función llamada Steffensen(f,a,tolerancia,errorfun,maxiter) para implementar el Método de Steffensen.**
**En este método las iteraciones siguen la fórmula de recurrencia**

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

Utiliza una estructura *Xsteff* para almacenar la tabla.