



**UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO**  
**FACULTAD DE ESTUDIOS SUPERIORES ARAGÓN**



**MÉTODOS NUMÉRICOS**

**TAREA 4:**  
**Método de Newton-Raphson**

**GRUPO: 1404**

**ALUMNOS:**

**Orozco Pastor Samadhi**

**Torres Jiménez Edgar Iván**

**FECHA DE ENTREGA: 01 de diciembre del 2025**

## Código:

```
function [salEva, salEP] = newton_raphson(funcion,derivada,x0,EP)
%Función para realizar el método de Newton-Raphson para obtener la raíz de
%una función.
%salEva=xn ó raíz de la función
%salPE= Error porcentual de la raíz
%-----
%funcion=función a evaluar
%derivada= derivada de la función f(x), tanto la función como su derivada
%deben de ser una función anonima o HANDLE.
%x0: punto inicial
%EP: Error porcentual deseado
error_actual=100000000; %Le pongo un valor grande deliberadamente, para iniciar el ciclo.
while error_actual > EP
    if derivada(x0)==0 % Para evitar una indeterminación por dividir entre cero
        error('La derivada es cero. No se puede continuar con el método. :(');
    end
    x1 = x0 - funcion(x0) / derivada(x0); % Calculo de la nueva xr.
    error_actual = abs((x1 - x0) / x1) * 100;
    x0 = x1; %Se actualiza el nuevo punto para la próxima iteración.
end
salEva = x1;
salEP = error_actual;
end
```

## Capturas de pantalla del programa

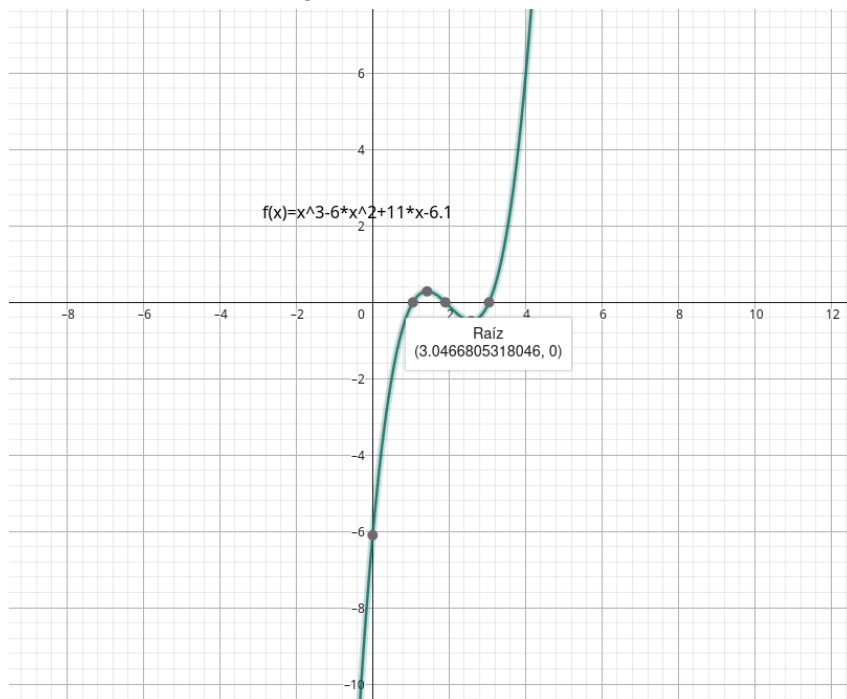
```
1 function [salEva, salEP] = newton_raphson(funcion,derivada,x0,EP)
2 %Función para realizar el método de Newton-Raphso para obtener la raíz de
3 %una función.
4
5 %salEva=xn ó raíz de la función
6 %salPE= Error porcentual de la raíz
7 %-----
8 %funcion=función a evaluar
9 %derivada= derivad de la función f(x), tanto la función como su derivada
10 %deben de ser una función anonima o HANDLE.
11 %x0: punto inicial
12 %EP: Error porcentual deseado
13
14 error_actual=1000000000; %Le pongo un valor grande deliberadamente, para iniciar el ciclo.
15 while error_actual > EP
16
17     if derivada(x0)==0 % Para evitar una indeterminación por dividir entre cero
18         error('La derivada es cero. No se puede continuar con el método. :(');
19     end
20
21
22     x1 = x0 - funcion(x0) / derivada(x0); % Calculo de la nueva xr.
23     error_actual = abs((x1 - x0) / x1) * 100;
24     x0 = x1; %Se actualiza el nuevo punto para la próxima iteración.
25 end
26
27 salEva = x1;
28 salEP = error_actual;
29
30 end
```

## Captura de pantalla del funcionamiento del programa:

Para la función  $f(x)=x^3-6x^2+11x-6.1$   
tenemos que su derivada es  $3x^2-12x+11$

```
f =  
function_handle with value:  
    @(x)x^3-6*x^2+11*x-6.1  
>>  
>> df= @(x) 3*x^2-12*x+11  
df =  
function_handle with value:  
    @(x)3*x^2-12*x+11  
>> [raiz, error]=newton_raphson(f,df,3,1)  
newton_raphson is not found in the current folder or on the MATLAB path, but  
exists in:  
    /home/samzhand/Matlab/2026-1MN/5/codigo  
Change the MATLAB current folder or add its folder to the MATLAB path.  
⚡ Explain Error  
>> cd /home/samzhand/Matlab/2026-1MN/5/codigo  
>> [raiz, error]=newton_raphson(f,df,3,1)  
  
raiz =  
  
    3.0467  
  
error =  
  
    0.1085  
  
>>
```

## Comprobación en Geogebra:



Para la función:  $g(x)=\ln(x^2+1)-\cos(\pi x)$

Tenemos que la derivada es:  $g'(x)=\pi \sin(\pi x)+2(x/(x^2+1))$

```
>> g=@(x)log(x^2+1)-cos(pi*x)

g =

function handle with value:

    @(x)log(x^2+1)-cos(pi*x)

>> dg=@(x)pi*sin(pi*x)+2*(x/(x^2+1))

dg =

function handle with value:

    @(x)pi*sin(pi*x)+2*(x/(x^2+1))

>> [raiz, error]=newton_raphson(g,dg,1.2,1/1000000)

raiz =

    -0.4427

error =

    4.7020e-12
```

Comprobación en Geogebra:

