NewtonRaphson

May 21, 2020

1 Método de Newton-Raphson

prof. Doherty Andrade – www.metodosnumericos.com.br

O Método de Newton-Raphson é um método de passo 1 e com ordem de convergëncia $ p=2$ muito utilizado para determinar numericamente a solução de equações não lineares $f(x) = 0$. A ordem de convergëncia $ p=2$ significa que é muito rápido na convergência.

É um método que exige que a função f seja de classe C^2 e utiliza derivada f'. Dado uma aproximação inicial x_0 o método é dado por:

$$_{+1} = \frac{()}{()}, n \geq 0.$$

```
In [1]: def newton(f,Df,x0,epsilon,max_iter):
            '''Calcula a solução aproximada de f(x)=0 pelo método de Newton.
            Parâmetros
            ----------
            f : função
                Função para a qual estamos procurando uma solução de f(x)=0.
            Df : função
                Derivada de f(x).
            x0 : número
                Candidato inicial para uma solução de f(x)=0.
            epsilon : número
                O Critério de parada é abs(f(x)) < epsilon.
            max_iter : inteiro
                Número máximo de iterações do método de Newton.
            Retorno
            -------
            xn : número
                Implementação do método de Newton: calcula a aproximação linear de f(x) em  xn
                intercepto x pela formula
                    x = xn - f(xn)/Df(xn)
                Continua até que  abs(f(xn)) < epsilon e retorna xn.
                Se Df(xn) == 0, retorna None.  Se o número de iterações excede o número
                max_iter, então o procedimento retorna None.

            Exemplos
```

```
    --------
    >>> f = lambda x: x**2 - x - 1
    >>> Df = lambda x: 2*x - 1
    >>> newton(f,Df,1,1e-8,10)
    Encontra a solução após 5 iterações.
    1.618033988749989
    '''
    xn = x0
    for n in range(0,max_iter):
        fxn = f(xn)
        if abs(fxn) < epsilon:
            print('A solução aproximada encontrada após',n,'iterações é:')
            return xn
        Dfxn = Df(xn)
        if Dfxn == 0:
            print('A derivada é nula. O método falha.')
            return None
        xn = xn - fxn/Dfxn
    print('O número máximo de iterações foi exceido. Nenhuma solução foi encontrada.')
    return None
```

1.1 Exemplo 1: Determinar a solução de $x^2 - x - 1 = 0$ com chute inicial $x_0 = 1.0$.

```
In [3]: f = lambda x: x**2 - x - 1
        Df = lambda x: 2*x - 1
        newton(f,Df,1,1e-8,10)
```

```
A solução aproximada encontrada após 5 iterações é:
```

```
Out[3]: 1.618033988749989
```

1.2 Exemplo 2: Determinar a solução de $x^3 - x^2 - 1 = 0$ com chute inicial $x_0 = 1.0$.

```
In [4]: f = lambda x: x**3 - x**2 - 1
        Df = lambda x: 3*x**2 - 2*x
        approx = newton(f,Df,1,1e-10,10)
        print(approx)
```

```
A solução aproximada encontrada após 6 iterações é:
1.4655712318767877
```

1.3 Exemplo 3: Determinar a solução de $x - \cos(x) = 0$ com chute inicial $x_0 = 0.5$.

```
In [5]: import math as m
        f = lambda x: x-m.cos(x)
        Df = lambda x: 1+m.sin(x)
        approx = newton(f,Df,0.5,1e-10,10)
        print(approx)
```

```
A solução aproximada encontrada após 4 iterações é:
0.7390851332151607
```

```
In [ ]:
```