Método de Newton-Raphson

José Ramón Pérez Navarro

September, 2019

0.1. Definición

El método de NewtonRaphson se basa en una aproximación por incrementos equivalente a una expansión en series de Taylor hasta la primera derivada.

Los métodos de cálculo estándar se utilizan para encontrar un método con rapidez de convergencia satisfactorio. Si la iteración ha alcanzado el punto x_n, entonces se requiere un incremento Δx_nque tomara el proceso hasta el punto de solución x$_*$. Si se hace la expansión de $f(x^*)$ en series de Taylor se tiene que:

$$0 = f(x^*) = f(x_n + \Delta x_n) = f(x_n) + \Delta x_n f'(x_n) + \frac{(\Delta x_n)^2}{2!} f''(x_n) + ...$$

Si la distancia Δx_nentre el punto de la iteración actual y la solución verdadera es suficientemente pequeña, entonces, tomando los dos primeros términos del lado derecho de la ecuación anterior, se obtiene:

$$0 \approx f(x_n) + \Delta x_n f'(x_n)$$

Asi, despejando Δx_n se llega a la expresión

$$\Delta x_n = \approx -\frac{f(x_n)}{f'(x_n}$$

Teorema

Suponiendo que f tiene segunda derivada continua y sea x_* tal que $f(x_*)$ =0 y $f'(x^*) \neq 0$. Si x_0 es lo suficientemente cercana a x*,la sucesion$x_{k_{k=0}^{\infty}}$generada por el metodo de Newton-Raphson converge a x^* con orden de convergencia de al menos 2.

Programa en Matlab

```
%  Programa principal para determinar los cruces por cero de una función continua.
%  Utilizando el método de Newton-Raphson, determinar los cruces por cero de la
%  función.
%  fx = 1 + 2.*x - 3.*x.^2.*exp(-x) + 2.*x.^3.*sin(x).*exp(-x./5) dentro del
%  intervalo [4,20]. Utilizar un error relativo de 1e-6. r=0;
% Inicia el contador de cruces por cero. for k = 4:20    l = k+1;
% Evaluación de la función en el punto k.    gk = 1 + 2.*k - 3.*k.^2.*exp(-k) + 2.
% Evaluación de la función en el punto l=k+1.    gl = 1 + 2.*l - 3.*l.^2.*exp(-l)
% Condicional que marca el subintervalo donde se encuentra un cruce por cero.    i
% Metodo de Newton-Raphson.        [Cero,Mat] = Newton-Raphson(k);        r = r
% Contador de cruces por cero.        Cruce(r) = Cero;
% Almacena los cruces por cero.        dr = length(Mat(:,1));
% Número de iteraciones para encontrar el cruce
% por cero.        dc = length(Mat(1,:));
% Número de variables por almacenar.        M(r,1:dr,1:dc) = Mat;
% Matriz que almacena todas las iteraciones de
% todos los cruces por cero.    end end M1(:,:) = M(1,:,:);
% Iteraciones del primer cruce por cero. M2(:,:) = M(2,:,:);
% Iteraciones del segundo cruce por cero. M3(:,:) = M(3,:,:);
% Iteraciones del tercer cruce por cero. M4(:,:) = M(4,:,:);
% Iteraciones del cuarto cruce por cero. M5(:,:) = M(5,:,:);
% Iteraciones del quinto cruce por cero.
```

Función

```
%  Función del método de Newton-Raphson para cálculo de cruces por cero de una
%  función no lineal que se mueve en plano real.
% %  El método calcula un cruce por cero.
% %  La función se llama de la siguiente manera.
% %        [Cero,Mat]=Newton-Raphson(a)
% %  Entradas:
%         a  -- Punto inicial del método.
% %  Salida:
%         Cero -- Valor de la variable para la cual la magnitud de la función es c
```

```
%         o menor a una tolerancia especificada previamente.
%         Mat  -- Vector que contiene todas las iteraciones hasta convergencia.
% function[Cero,Mat] = NewtonRaphson(a); Err = 1;
% Inicializa el error para ingresar al ciclo iterativo. tol = 1e-12;
% Tolerancia especificada para la convergencia. c   = 0;
% Inicializa el contador de iteraciones. while Err > tol & c < 20
% Valor de la función en el punto inicial.   fa  = 1 + 2.*a - 3.*a.^2.*exp(-a) + 2
% Valor de la derivada de la función en el punto inicial.   fpa = 2 - 6.*a.*exp(-a
% Cálculo del nuevo valor de x dado por el método de Newton-Raphson.
 xn = a - fa/fpa;
 % Valor de la funcion en el nuevo punto.   fxn = 1 + 2.*xn - 3.*xn.^2.*exp(-xn) +
 % Contador de iteraciones para no dejar en un ciclo el programa en caso de alguna
 % inconsistencia.   c = c + 1;
 % Matriz que almacena los resultados de cada iteración.   Mat(c,:) = [a fa fpa];
 %  Cruce por cero que determina el método de Newton-Raphson. Cero = xn;
```