Instituto Tecnológico de Costa Rica

Área Académica Ingeniería en Computadores

II Semestre 2020

Análisis Numérico para Ingeniería (CE3102)

Tarea #2

Método de Newton-Raphson, para Solución de Sistemas de Ecuaciones no Lineales

Estudiantes:

Fiorella Delgado León - 2017121626

Cristian Marin Murillo - 2016134345

Randy Martínez Sandí - 2014047395

Karla Michelle Rivera Sánchez - 2016100425

Profesor:

Ing. Juan Pablo Soto Quiroz.

Fecha de entrega:

Domingo 29 de noviembre del 2020

1. Introducción

Para resolver un sistema de ecuaciones no lineales de forma iterativa, se puede hacer uso del Método Newton-Raphson, el cual es el más común y hace uso de la Matriz Jacobiana. Este método utiliza valores iniciales para resolver un sistema de n ecuaciones con n incógnitas, las cuales no resuelven el sistema, y de forma iterativa actualiza los valores hasta cumplir con el criterio de parada definido por la tolerancia dada.

El método aproxima las soluciones de la ecuación

$$f(x) = 0; \text{ donde } f : \mathbb{R}^n \rightarrow \mathbb{R}^m \text{ y } x \in \mathbb{R},$$

donde $f(x) = 0$ no representa un sistema de ecuaciones lineales.

2. Matriz Jacobiana o jacobiano

Es una matriz $(c) \in \mathbb{R}^{nxn}$ definida mediante la siguiente ecuación:

$$[\ (c)]_{i,j} = \frac{\partial f_i}{\partial x_j}(c),$$

donde cada $f_j : \mathbb{R} \to \mathbb{R}$ es una función definida y continua en $= (f_1, \ldots, f_n) : \mathbb{R}^n \to \mathbb{R}^n$, cuyas primeras derivadas parciales $\frac{\partial f_i}{\partial x_j}$ existen en $c = (x_1, \ldots, x_n)^t \in \mathbb{R}^n$ para $i, j = 1, 2, \ldots, n$.

3. Método de Newton para Sistemas de Ecuaciones no Lineales

Este Método iterativo se define de la siguiente manera:

$$\begin{cases} x_{k+1} = x_k - [\mathcal{J}_{\mathbf{f}}(x_k)]^{-1} \cdot \mathbf{f}(x_k) \\ x_0 \in \mathbb{R}^n : \text{Vector inicial} \end{cases}$$

donde (x_k) es la matriz jacobiana, la cual es invertible para todo $k = 0, 1, 2, \ldots$

Para este método se debe calcular:

$$y = [\ (x)]^{-1} \cdot (x_k)$$

Teniendo entonces la ecuación del Método de esta forma:

$$x_{k+1} = x_k - y$$

Computacionalmente, vamos a considerar como criterio de parada: $\| F(x^{(k)}) \|_2 < tol$, entonces el $x^{(k)}$ que cumpla el criterio es la solución. Se tiene que tol es una tolerancia dada como parámetro de entrada.

También se debe tomar en cuenta que x_0 no contiene los valores solución del sistema de ecuaciones, teniendo así $f(x_0) \neq 0$.

4. Valores iniciales

Para la implementación computacional de este Métodos, se definen los siguientes valores iniciales.

1. $x^{(0)} \in \mathbb{R}^n$: Vector inicial.
2. $= (f_1, \ldots, f_n)$: Vector *string*, se tiene que cada $f_i(x_1, \ldots, x_n) : \mathbb{R}^n \to \mathbb{R}$ es función no lineal.
3. $x = (x_1, \ldots, x_n)$: Vector *string*, incógnitas a utilizar.
4. $tol > 0$, tolerancia para el criterio de parada $\| F(x^{(k)}) \|_2 < tol$
5. $iterMax > 0$, número de iteraciones máximas.

5. Pseudocódigo

El pseudocódigo del método de Newton-Raphson, se puede describir de la siguiente manera:

Parámetros de entrada:

- Vector inicial: $x^{(0)} = (x_1, x_2, \ldots, x_n)$.
- Vector tipo string (ecuaciones): $= (f_1, \ldots, f_n)$.
- Vector tipo string (incógnitas): $x = (a_1, a_2, \ldots, a_n)$.
- Tolerancia: tol .
- Número de iteraciones máximas: $iterMax$.

Parámetros de salida:

- Vector resultado: $x^{(k)} = (x_1, x_2, \ldots, x_n)$.
- Cantidad de iteraciones que realizó: Vector inicial: k.
- Error generado: e_k .
- Una gráfica de iteraciones vs errores.

Paso 0: Calcular la Matriz Jacobiana usando la siguiente ecuación:

$$[\ (c)]_{i,j} = \frac{\partial f_i}{\partial x_j}(c)$$

Paso 1: Inicializar el contador $k = 0$, así como el valor del vector $x^{(k)}$.

Paso 2: Calcular el valor de x^{k+1} usando la siguiente ecuación:

$$x_{k+1} = x_k - [\ (x)]^{-1} \cdot\ (x_k)$$

Paso 3: Verificar si se cumple o no el criterio de parada de las iteraciones, usando el valor anteriormente calculado, mediante la siguiente fórmula:

$$\| F(x^{(k)}) \|_2 < tol$$

o que $k = iterMax$ y comprobar:

- Si cumple, entonces el vector x_k contiene la mejor aproximación de la solución del sistema.
- Si no cumple, continuar con el Paso 4.

Paso 4: Incrementar en 1 el valor del contador k, y volver al Paso 2.