Pontificia Universidad Javeriana de Cali

Facultad de Ingeniería de sistemas y computación

Punto Flotante, Gauss y Gauss Jordan

Practica Computación Cientifica

Autores:
Andrés Felipe Delgado y Ana Maria García

Febrero 28 del 2020

1 Introducción

En esta primera practica de computación científica se va a programar en Matlab algoritmos iterativos rápidos y suficientemente exactos para graficar sistemas de punto flotante normalizados y solucionar sistemas de ecuaciones lineales cuadrados mediante el método de eliminación Gaussiana y de Gauss-Jordan.

A continuación se va a especificar la forma en que operan los algoritmos implementados en los diversos casos.

2 Gráfica sistema punto flotante normalizado

Para la representación gráfica del sistema de punto flotante se recibió como entrada del algoritmo la base (B) , precisión (t) y el rango del exponente [lower (L) ,Upper(U)], todos de tipo entero.

Con estos datos se calculó la cantidad de números flotantes que el sistema puede representar (N) mediante la siguiente ecuación:

$$N = 2(B-1)B^{t-1}(U-L+1)+1 \tag{1}$$

Luego calculamos el número más grande que se puede representar en el sistema (OFL):

$$OFL = B^{U+1}(1-B^{-t}) \tag{2}$$

Finalmente, calculamos el número más pequeño que puede representar el sistema (UFL):

$$UFL = B^{L} \tag{3}$$

Con esta información generamos un vector con todos los números pertenecientes al sistema que luego graficamos. A continuación se presenta el funcionamiento del algoritmo con dos ejemplos distintos.

2.1 Funcionamiento algoritmo

- ejemplo 1:

  Dado B = 2, t = 3, L = -1, U = 1 , calculamos con (1) que N = 25, con (2) que OFL = 3.5 y con (3) UFL = 0.5.

  Por tanto el sistema soporta 25 números donde el menor es 0.5 y el mayor es 3.5. Lo que quiere decir que son 12 números positivos, 12 negativos y el cero.

Lo primero es calcular y guardar las potencias de 2 entre UFL y OFL teniendo en cuenta la potencia que le sigue a OFL en caso de que OFL no sea potencia de 2.

En este caso el algoritmo arroja en el vector r las siguientes potencias de 2:

r = [1,2,4]

ahora calculamos las divisiones dentro de los intervalos de r, como se encontraron 3 potencias de 2 y son 12 números positivos los que se deben representar, se hace:

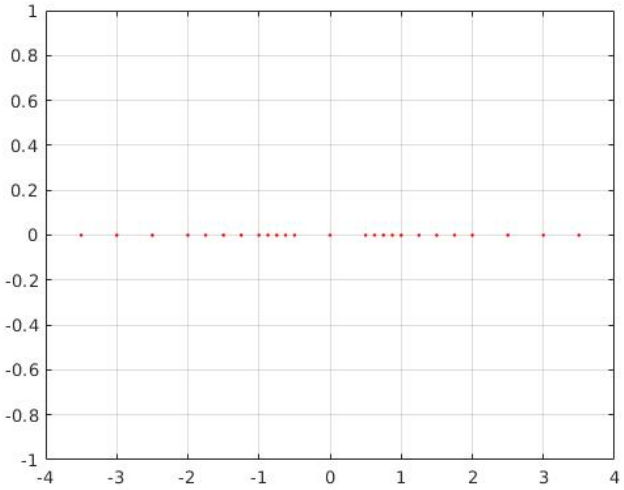
$$numerodivisiones = (N/2) - 1/length(r) = 12/3 = 4 \qquad (4)$$

Ahora realizamos las divisiones entre los distintos intervalos del sistema.

entre 0.5 y 1 : [0.625,0.75,0.875]
entre 1 y 2 : [1.25,1.5,1.75]
entre 2 y 4 : [2.5,3,3.5]

Con esto tenemos los 12 números positivos del sistema los cuales son:
0.5, 0.625, 0.75, 0.875, 1, 1.25, 1.5, 1.75, 2, 2.5, 3 y 3.5

Finalmente añadimos las versiones negativas de estos 12 números, el cero y gráficamos. La gráfica que se crea después de ejecutar el algoritmo es la siguiente.



- ejemplo 2:

  Dado B = 2, t = 4, L = -1, U = 1 , calculamos con (1) que N = 49, con (2) que OFL = 3.75 y con (3) UFL = 0.5.

Por tanto el sistema soporta 49 números donde el menor es 0.5 y el mayor es 3.75. Lo que quiere decir que son 24 números positivos, 24 negativos y el cero.

Lo primero es calcular y guardar las potencias de 2 entre UFL y OFL teniendo en cuenta la potencia que le sigue a OFL en caso de que OFL no sea potencia de 2.

En este caso el algoritmo arroja en el vector r las siguientes potencias de 2:

r = [1,2,4]

ahora calculamos las divisiones dentro de los intervalos de r, como se encontraron 3 potencias de 2 y son 24 números positivos los que se deben representar se hace:

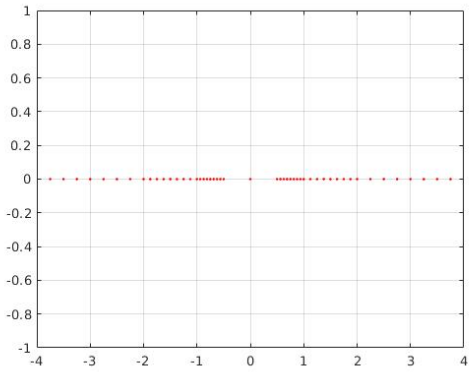
$$numerodivisiones = (N/2) - 1/length(r) = 24/3 = 8 \tag{5}$$

Ahora realizamos las divisiones entre los distintos intervalos del sistema.

entre 0.5 y 1 : [0.5625,0.625,0.6875,0.75,0.8125,0.875,0.9375]
entre 1 y 2 : [1.125,1.25,1.375,1.5,1.625,1.75,1.875]
entre 2 y 4 : [2.25,2.5,2.75,3,3.25,3.5,3.75]

Con esto tenemos los 24 numeros positivos del sistema los cuales son:
0.5, 0.5625,0.625,0.6875,0.75,0.8125,0.875,0.9375,1 1.125,1.25,1.375,1.5,1.625,1.75, 1.875,2 2.25,2.5,2.75,3,3.25,3.5,3.75

Finalmente añadimos las versiones negativas de estos 24 números, el cero y graficamos. La gráfica que se crea después de ejecutar el algoritmo es la siguiente.



3 sistema de ecuaciones lineales

En notación matricial un sistema de ecuaciones lineales algebraicas tiene la forma

$$AX = b \tag{6}$$

Donde A es una matriz de m*n (en esta practica se trabajará solo con matrices cuadradas donde m=n) b es un vector columna de m filas, y X la solución desconocida de tamaño n.

Existen diversas formas de solucionar este tipo de sistemas lineales, en esta practica nos centraremos en dos, el método de eliminación Gaussiana y el método de eliminación Gauss-JOrdan que es una variante del primero.

En ambos vamos a hacer uso de la mejor solución computacionalmente hablando que consiste en resolver el sistema mediante pre-multiplicaciones por matrices de eliminación fundamentales y permutaciones.

3.1 Matrices de eliminación elementales

Es una matriz que busca aniquilar todos los valores de la matriz A debajo de un pivote.

El pivote es el número de la columna que pertenece a la diagonal principal de la matriz y es diferente de cero, en caso de que el pivote sea cero se realiza una permutación de filas.

Si generamos una matriz de eliminación fundamental por cada columna de A y multiplicamos de la siguiente forma de manera iterativa:

$$MAX = Mb \tag{7}$$

Donde M es la matriz de eliminación fundamental, llegaremos a que A va a quedar de la siguiente forma:

$$\begin{Bmatrix} A11 & A12 & A13 \\ 0 & A22 & A23 \\ 0 & 0 & A33 \end{Bmatrix}$$

Este tipo de Matriz es conocida como tringular superior y llevar a A a esta forma es el primer paso para solucionar el sistema con el metodo de eliminación Gaussiana como se explica a continuación

3.2 Solución sistema de ecuaciones lineales con eliminación Gaussiana

Una vez A sea triangular superior existe una solución iterativa bastante sencilla para el sistema:

$$Xn = bn/Ann \tag{8}$$

$$Xi = bi - \sum_{j=i+1}^{n} A_{ij} X_j$$

$$Xi = Xi/Aii \tag{10}$$

Donde i = n-1 ,...., 1

3.3 Ejemplo solución Eliminación Gaussiana

Dado:

$$\text{A} = \begin{Bmatrix} 2 & -3 & 1 \\ -2 & 0 & -2 \\ 3 & -2 & -3 \end{Bmatrix}$$

$$\text{X} = \begin{Bmatrix} x \\ y \\ z \end{Bmatrix}$$

$$\text{b} = \begin{Bmatrix} -2 \\ 0 \\ 4 \end{Bmatrix}$$

Teniendo en cuenta la ecuación (6), al despejar X que son los valores que queremos hallar para solucionar el sistema, tenemos:

$$X = A^{-1} * b \tag{11}$$

Donde A elevado a la -1 indica la inversa de la matriz A. Esta puede calcularse de diversas formas. Finalmente, con el resultado de la matriz A inversa, esta se multiplica por el vector b y se obtiene los valores de X, en este caso:

$$\text{X} = \begin{Bmatrix} 1 \\ 1 \\ -1 \end{Bmatrix}$$

4 Solución sistema de ecuaciones lineales con Gauss-Jordan

La segunda forma de solución que se abarco en esta practica como ya mencionamos es una variacion de la primera.

El principal cambio radica en que en esta tecnica buscamos llevar a A a ser una matriz diagonal:

$$\begin{Bmatrix} A11 & 0 & 0 \\ 0 & A22 & 0 \\ 0 & 0 & A33 \end{Bmatrix}$$

Esto lo logramos generando matrices de eliminación fundamental que no solo aniquilen la parte inferior del pivote sino también la superior.

La solución iterativa en este caso es mas sencilla:

$$Xi = Xi/Aii \tag{12}$$

Donde i = 1 ,...., n

4.1 Ejemplo solución Gauss-Jordan

Dado el sistema de ecuaciones

$$2x_1 + 4x_2 - 2x_3 = 2$$
$$4x_1 + 9x_2 - 3x_3 = 8$$
$$-2x_1 - 3x_2 + 7x_3 = 10$$

$$\text{A} = \begin{Bmatrix} 2 & 4 & -2 \\ 4 & 9 & -3 \\ -2 & -3 & 7 \end{Bmatrix}$$
$$\text{X} = \begin{Bmatrix} x1 \\ x2 \\ x3 \end{Bmatrix}$$
$$\text{b} = \begin{Bmatrix} 2 \\ 8 \\ 10 \end{Bmatrix}$$

Teniendo en cuenta la ecuación (6), al despejar X que son los valores que queremos hallar para solucionar el sistema, tenemos:

$$X = A^{-1} * b \tag{13}$$

Donde A elevado a la -1 indica la inversa de la matriz A. Esta puede calcularse de diversas formas. Finalmente, con el resultado de la matriz A inversa, esta se multiplica por el vector b y se obtiene los valores de X, en este caso:

$$\text{X} = \begin{Bmatrix} -1 \\ 2 \\ 2 \end{Bmatrix}$$

5 Conclusiones

En este trabajo hemos podido llegar a varias conclusiones en diferentes aspectos. Como podemos observar en la sección 2.1, ejemplo 2, al incrementar el t, la exactitud del sistema creció directamente proporcional, duplicándose al igual que t, pues se pudieron representar el doble de números que en el ejemplo 1

donde t era 2. Por otro lado, en cuanto a la herramienta utilizada Matlab, se pudo evidenciar que las operaciones recursivas para realizar ciertos cálculos son mucho menos costosas que calcular la inversa de una matriz de la forma convencional, la cual sería la manera más trivial de solucionar el problema.

En conclusíon mediante esta practica logramos el objetivo principal de la computación cientifica que es encontrar y aplicar algortimos suficientemente rapidos y exactos para solucionar un problema.