Universidad Nacional Autónoma de México

Facultad de Estudios Superiores Acatlán

Ejercicio 6: Método de Gauss

Materia: Métodos Numéricos

Autor: Díaz Valdez Fidel Gilberto
Número de cuenta: 320324280

Octubre 2023

1 Propósito

Desarrollar la capacidad de reconocer y estar familiarizado con los patrones en el manejo de los subíndices para poder implementar el método de Gauss en un programa debido a la gran eficacia de este.

2 Instrucciones

Elaborar en *pseudocódigo* un algoritmo para la obtención del determinante de matrices cuadradas por el método de eliminación gaussiana, triangulando una matriz dada.

- Detallado
- Manejo claro de los índices
- Identificar matrices singulares.

3 Planteamiento

Lo ideal antes de hacer operaciones es revisar que efectivamente la matriz a tratar tiene solución, es decir, es *no singular* y para esto existen varios métodos o formas para comprobar si la matriz cumple ciertas características. Si cumple estas signfica que es *singular*:

- Dos columnas/renglones iguales entre sí.
- Renglón/columna múltiplo constante de otra.
- Cualquier renglón/columna de la matriz es el vector 0.

Una condición interesante que nos permite conocer si una matriz es invertible es si es *estrictamente dominante diagonalmente*, pero si esto no se cumple no implica que no sea invertible, funciona para conocer si una matriz tiene solución. Esta se explica como si en cada miembro de la diagonal en valor absoluto es mayor a la suma de los elementos restantes del renglón en valor absoluto.

También es reconocible con la siguiente expresión:

$$|a_{i,j}| \geq \sum_{j=1, i \neg j}^{n} |a_{i,j}|$$

Por lo tanto lo ideal será primero revisar si es *estrictamente dominante diagonalmente*, ya que si lo es podemos estar seguros de que es invertible, si fuese el caso en el que no lo es, bastará con revisar si se tiene algún cero en la diagonal al triangulizar la matriz para saber que su determinante será igual a cero, esto debido que al triangulizar la matriz la manera de calcular el determinante es a través de multiplicar cada componente de la diagonal entre sí.

Ejemplo de una matriz ya triangularizada:

$$A = \begin{bmatrix} 5 & 2 & 3 & -1 \\ 0 & \frac{16}{5} & \frac{-6}{5} & \frac{7}{5} \\ 0 & 0 & \frac{9}{8} & \frac{-5}{15} \\ 0 & 0 & 0 & \frac{65}{8} \end{bmatrix}$$

Por practicidad no explicare la forma para llegar a la matriz triangulada a través de la eliminación gaussiana, solo se hará un repaso general.

Se busca que todo lo que este debajo de la matriz triangular sean valores de cero y esto se logrará haciendo operaciones aritméticas donde el elemento de la matriz donde i, j son iguales, es decir el elemento que se encuentran en la diagonal, será utilizado para hacer cero los elementos de abajo de este, como se muestra en el ejemplo.

4 Construcción

Primero se creará el pseudocódigo para llenar la matriz por el usuario:

```
i,j ENTERO
ESCRIBIR("Dame el numero de columnas")
LEER(columnas)
filas = columnas


matriz = ^^ENTERO
Crea (matriz) \\Con el número de filas
Para (i desde 1 hasta i<= filas con paso 1)
   CREA(matriz[i])
FINPARA


PARA(i desde 1 hasta i<= filas con paso 1)
   PARA(j desde 1 hasta j <= columnas con paso 1)
      ESCRIBIR("Dame el valor a guardar")
      LEER(matriz[i][j])
   FINPARA
FINPARA
```

Se pregunta de cuántas columnas es la matriz y como la finalidad será encontrar la determinante, sabemos que esto es posible solo a tener una matriz cuadrada por lo que no permitimos que el usuario digite una matriz no cuadrada al no preguntar el número de filas sino más bien haciéndolo igual al de columnas.

Continuamos creando la matriz y cada uno de sus celdas de memoria, primero se asigna las filas y después dentro de un ciclo se comienzan a apartar los cachos de memorias para cada columna de cada fila. Por último se recorre la matriz para llenarla con los datos deseados.

Una vez con la matriz inicializada se revisará que esta sea *estrictamente diagonalmente dominante*:

```
diagonal, i, j, control, temp, total ENTERO
i = 1
j = 1
control = 1
temp = 0
total = 0
MIENTRAS(i<= filas Y control != 0)
   PARA(j desde 1 hasta j <= columnas con paso 1)
      SI j = i
         diagonal = matriz[i][j]
      SINO
         SI matriz[i][j] < 0
            temp = matriz[i][j] * -1
         SINO
            temp = matriz[i][j]
         FINSI


         total = temp + total
      FINSI
   FINPARA

SI total < diagonal
   control = 0
SINO
   control = 1
FINSI
i = i +1
FINMIENTRAS
```

Este fragmento de pseudo código hace que se recorra la matriz siempre y cuando no se exceda el número de filas, es decir no se salga de la matriz y la variable control sea distinta de 0 Se comenzará a recorrer la matriz y cuando se encuentre el valor de la diagonal se guardará en la variable diagonal, sino se está en la diagonal, se transformara su valor a positivo, se guardará en la variable temp para que después sea sumado con total que será la variable que contenga la suma de todos los elementos de un renglón exceptuando la diagonal.

Una vez terminado con el renglón se revisa que la diagonal sea mayor a la suma de los elementos restantes del renglón, si la diagonal es menor la variable

control será igual a cero sacando del recorrido al programa pues ya sabemos que no es *EDD* pero si si lo es, se continúa el recorrido, por esta razón a i se le suma una unidad, para llegar a la siguiente fila.

Gracias a la variable *control* ahora podemos saber si es *EDD* o no, lo que continuará es, si es *EDD* realizar el método de eliminación gaussiana sin preocupación ya que es seguro que esa matriz tiene inversa y por lo tanto determinante distinto de cero. Por otro lado si no es *EDD* deberemos realizar la eliminación gaussiana pero revisando que no exista ningún cero en la diagonal principal, ya que como ya vimos en el planteamiento esto significa que se trata de una matriz singular.

```
FLOTANTE factor, det = 1
ENTERO a, b, singular
MIENTRAS i <= filas Y singular != 0
   PARA (a desde i+1 hasta a<=filas con paso 1)
      SI matriz[a][i]*matriz[i][i] < 0
         factor = matriz[a][i]/matriz[i][i]
      SINO
         factor = (matriz[a][i]/matriz[i][i]) * -1
      FINSI
      PARA(b desde i hasta b<=columnas con paso 1)
         matriz[a][b] = factor * matriz[i][b]+ matriz[a][b]
         SI b = a
            SI matriz[a][i] == 0
               singular = 0
            FINSI
         FINSI
      FINPARA
   FINPARA
   i = i+1
FINMIENTRAS


PARA(i desde 1 hasta i<=filas con paso 1)
   det = matriz[i][i] * det
FINPARA
```

Este fragmento de pseudocódigo ya es la manera en que realizaremos la eliminación gaussiana y la obtención del determinante.

Primero se definen las variables que utilizaremos, como solo nos importa partir desde la diagonal principal pues esta es la que triangularizar a la matriz, basta con usar solo un índice que en este caso será i, así se moverá siempre en la diagonal principal ya que usaremos a *matriz* en términos de i en ambos campos, tanto en el de columna como el de fila. Será un ciclo mientras para poder tener una variable de control *singular* que nos alertará si en algún momento en la diagonal principal hay un cero, ya que si es así no tiene caso continuar pues

significa que el determinante es cero, cuando eso suceda se hará a *singular* igual a cero para que así se salga del ciclo MIENTRAS. Una vez ahí posicionado necesitamos movernos en las filas de abajo de la fila en la que estamos actualmente, es por esto que se define un ciclo para que estará una fila adelante y crecerá con una unidad a la vez, teniendo su cuestión de paro como cuando se es mayor al número de filas ingresado por el alumno.

Se revisa si el elemento de la matriz que desea hacer ceros debajo de él es del mismo signo, ya que si es así será necesario que el factor multiplicativo sea negativo para que la suma con el renglón que se quiere sea cero, realmente tenga ese resultado. El caso en el que los elementos del renglón son de diferente signo al del renglón que se busca hacer cero (al menos la columna debajo de elemento de la diagonal principal) es análogo para que el resultado sea cero.

Ya teniendo la variable *factor* configurada para que la suma con el renglón de abajo sea cero, se hará otro ciclo para hacer la operación de $factor_Mult(R_i)+R_a$ en cada columna, la variable b es la que se encarga de recorrer todas las columnas y asigna los nuevos valores dentro de este ciclo. Una vez realizada la operación se revisa si es valor está en la diagonal principal y si es así, si es diferente de cero ya que si no lo es significa que es singular y ya no tiene caso continuar el cálculo, en este caso se activa nuestra variable de control *singular* y se sale del ciclo, en general.

De no suceder lo anterior se continúa con el ciclo como si nada se mueve al siguiente renglón y se repite hasta que se acaben las filas con las que hacer operaciones, por último para obtener el determinante solo se multiplicará cada valor de la diagonal hasta que ya se multipliquen todos entre sí, esto nos dará el resultado de si es singular o no lo es.

5 Conclusión

El algoritmo no está en uno solo porque considero que esto le brinda mayor flexibilidad para codificar como sea necesario, si deseas que se diga que es *EDD* se puede usar la variable *control*, si se desea parar y no calcular el cero en caso de ser *singular* se puede hacer uso de la variable *singular*, incluso si se desea omitir el revisar si hay algún cero en la diagonal debido a que ya se comprobó antes que si es *EDD* y por lo tanto tiene solución, se puede hacer una estructura SI/SINO en donde se omita la condición de revisión según el valor de *control*.

En conclusión si me costo realizar este pseudocódigo, pero me alegro hacerlo porque considero que si me ayudo a tener una mejor noción sobre los índices y cómo utilizarlos a mi favor en la construcción de futuros programas, lo considero un buen inicio además de útil.