Informe Laboratorio 1 Computación Científica

Josué Peña A., Jeffrey García G.

Febrero 2020

Contents

1 Introducción **2**

2 Sistema de punto flotante **2**
2.1 Sistema de ejemplo 2
2.2 Sistema adicional 3

3 Método de Gauss **3**
3.1 Comparación de eficiencia frente a MATLAB 4

4 Sustitución Sucesiva **4**

5 Método de Gauss-Jordan **4**
5.1 Comparación de eficiencia frente a MATLAB 4

6 Conclusiones **5**

1 Introducción

En este primer laboratorio se realizan visualizaciones e implementaciones de algunos de los temas vistos en lo que va del curso. Específicamente, se realizó la función *floating_point.m* en MATLAB que genera la visualización de un sistema de punto flotante graficando cada número representable como un punto. Con esta función se realizaron dos pruebas diferentes: una construyendo el sistema de ejemplo visto en clase, y otro sistema adicional propuesto.

El resto de temática implementada es sobre la resolución de sistemas de ecuaciones a través de matrices de eliminación con el método de Gauss y Gauss-Jordan. Se implementaron tres funciones, *gauss.m*, *gauss_jordan.m* y *suc_sust.m* (esta última correspondiendo al método de sustitución sucesiva) y se realizaron pruebas de comparación de velocidad con respecto a las funciones disponibles en la librería estándar de MATLAB.

2 Sistema de punto flotante

Para esta prueba no se probaron sistemas de contaran con una cantidad de números representables demasiado alta, ya que esto dificulta la visualización de cada punto individual y el cómo están distribuidos en la línea de los reales.

2.1 Sistema de ejemplo

Usando los mismos valores de los parámetros que se usan en el ejemplo visto en clase ($\beta = 2$, $t = 3$, $L = -1$, $U = 1$), se generó la siguiente gráfica de todos los 25 puntos que existen en el sistema. Este tiene como menor y mayor números positivos representables $UFL = 0.5$ y $OFL = 3.5$.

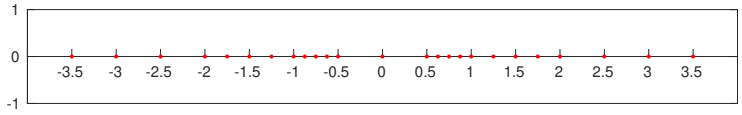

Figure 1: Gráfico sistema de ejemplo

2.2 Sistema adicional

Para probar el algoritmo desarrollado que gráfica sistemas de punto flotante, se decidió construir el sistema de punto flotante con $\beta = 2$, $t = 4$, $L = -1$ y $U = 1$. Este sistema adicional puede representar 49 números, con $UFL = 0.5$ y $OFL = 3.75$.

Como se evidencia en el gráfico, aumentar la precisión del sistema ha causado que el rango entre el OFL y el UFL tenga una mayor cantidad de puntos, permitiendo así que estos abarquen mucho más espacio que en el ejemplo anterior, logrando así una mejor *precisión* para los números que se pueden representar en el sistema.

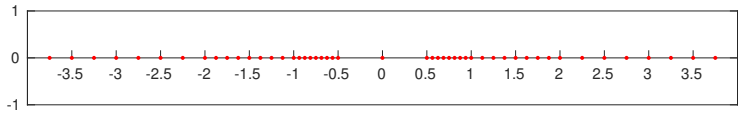

Figure 2: Gráfico sistema adicional

3 Método de Gauss

El objetivo de este algoritmo es realizar la reducción a una matriz triangular superior de un sistema de ecuaciones representado por una matriz nxn y un vector nx1 utilizando la eliminación gaussiana. El resultado de la reducción genera una matriz Ap, y un vector bp que son introducidos como parámetros en el algoritmo de sustitución sucesiva para obtener la solución al sistema.

En la implementación propuesta del algoritmo de Gauss primero es necesario comprobar si la matriz es cuadrada y su determinante es distinto a 0, es decir, que la matriz es no singular. Posteriormente se revisa por filas cada pivote, si el valor del pivote es 0, se realiza un intercambio de la linea actual con la siguiente. Luego para cada valor debajo del pivote se añade a la matriz de eliminación el valor que lo elimine (lo convierta a 0). Por último se realiza la multiplicación de la matriz de eliminación con la matriz y vector de entrada, y se repite este procedimiento para cada pivote.

3.1 Comparación de eficiencia frente a MATLAB

En esta prueba se utiliza el ejemplo dado en clase para realizar las comparaciones entre las dos funciones *gauss.m* y *suc_ sust.m*, frente al operador de MATLAB.

$$A = \begin{bmatrix} 2 & 4 & -2 \\ 4 & 9 & -3 \\ -2 & -3 & 7 \end{bmatrix}, b = \begin{bmatrix} 2 \\ 8 \\ 10 \end{bmatrix}$$

El tiempo medido en MATLAB con la herramienta *profile* para las dos funciones implementadas retorna un tiempo de 0.029 s para encontrar la solución al sistema de ejemplo.

Por otro lado, el operador "\" de MATLAB, utilizado para resolver sistemas de ecuaciones lineales en general, da un tiempo de 0.030 s. Esto es debido a que el operador está orientado para resolver una gama más amplia de sistemas de ecuaciones lineales incluyendo sistemas rectangulares y con otras propiedades o características, por lo tanto debe realizar más verificaciones para identificar qué tipo de sistema está trabajando.

4 Sustitución Sucesiva

El objetivo de este algoritmo es retornar el vector de valores para las distintas variables de un sistema reducido a una matriz triangular o a una matriz diagonal.

Para este algoritmo se empieza calculando los valores empezando desde la última fila de la matriz. Emulando el proceso de despeje manual, se calcula primero la parte independiente de la variable que se está calculando. Este valor se guarda y es dividido por el cociente de dicha variable. Los resultados se guardan en un nuevo vector.

5 Método de Gauss-Jordan

El objetivo del algoritmo es realizar la reducción a una matriz diagonal de un sistema de entrada, con matriz de tamaño nxn y un vector $nx1$. Donde el sistema reducido será resuelto por el algoritmo de sustitución sucesiva.

Este algoritmo se puede ver como una modificación al algoritmo de eliminación gaussiana utilizado para generar un sistema con una matriz triangular superior y su respectivo vector. La diferencia entre estos dos métodos es que en lugar de realizar la eliminación de los valores de las celdas debajo del pivote, este realiza la eliminación también por arriba, dejando como único valor de la columna al pivote, este proceso se repite para cada pivote en la matriz.

5.1 Comparación de eficiencia frente a MATLAB

En esta prueba se vuelve a utilizar el mismo sistema de ejemplo que en la sección anterior.

$$A = \begin{bmatrix} 2 & 4 & -2 \\ 4 & 9 & -3 \\ -2 & -3 & 7 \end{bmatrix}, b = \begin{bmatrix} 2 \\ 8 \\ 10 \end{bmatrix}$$

Se tiene que el tiempo de ejecución para este ejemplo fue de 0.036 s. En cambio, según el resultado que se tenía de la prueba anterior, con la operación por defecto de MATLAB el tiempo de ejecución fue de 0.030 s. Aquí podemos observar un aumento en el tiempo de ejecución del algoritmo desarrollado en el laboratorio, esto es debido a la funcionalidad adicional que debe realizar en comparación al algoritmo de eliminación gaussiana.

6 Conclusiones

De este laboratorio se puede concluir que las herramientas que brinda MATLAB para este tipo de trabajos de computación son muy completas y ayudan a mejorar la productividad en el desarrollo de algoritmos que requieren de este tipo de procedimientos. Además de eso, MATLAB facilita el manejo de operaciones con matrices y operaciones matemáticas gracias a su amplia gama de funciones y estructuras de datos.

Se pudo observar que las funciones y operaciones de MATLAB están optimizadas y son lo más generales posibles para abarcar cualquier tipo de caso, como se puedo ver evidenciado en el ejemplo del operador "\". Eso si, tal generalidad trae como consecuencia una pérdida de rendimiento para instancias concretas de los sistemas de ecuaciones.

Tal hecho resalta las decisiones que se deben tomar a la hora de implementar algoritmos. En ciertos contextos, la eficiencia es lo más importante, como en el software de un sistema de empotrado. Por tal razón es más provechoso implementar algoritmos que aprovechen y abarquen sólo las características del conjunto de problemas particulares que este debe computar.

Pero, en otros contextos como en el de MATLAB, se busca tener funciones y algoritmos que impulsen la productividad, sin preocuparse mucho de la eficiencia. Es por esto que tales implementaciones deben ser lo más generales posibles.