Laboratorio N°1 - Algoritmos Numéricos

Jennifer Andrea Velozo Bruna
Programación y análisis de algoritmos numéricos en Matlab

27 de mayo de 2021

Resumen

El presente documento muestra un análisis de métodos numéricos para ecuaciones no lineales y para sistemas de ecuaciones no lineales, programados en Matlab. Dicho análisis contempla la comparación de los errores obtenidos y la comparación de los costos temporales y espaciales en cada método. Cabe destacar que tanto la implementación de los métodos como las características del computador donde se ejecute el programa, podría afectar en los resultados obtenidos.

1. Introducción

Un método numérico es la aplicación de cálculos y álgebra que se llevan a una herramienta computacional informática, como por ejemplo Matlab, la cual es utilizada para el desarrollo de este laboratorio.

El presente documento tiene como objetivo mostrar el desarrollo y análisis del laboratorio N°1 de la asignatura Algoritmos Numéricos.

La experiencia se divide en dos partes:

1. Programar los métodos de resolución de ecuaciones no lineales y comparar los errores obtenidos y los costos temporales y espaciales.

2. Programar los métodos de resolución de sistemas de ecuaciones, tanto iterativos como directos, a partir de las matrices entregadas, y comparar sus costos temporales y espaciales.

2. Metodología

Los métodos para resolver las ecuaciones no lineales son los siguientes:

- Método de la Bisección [1]
- Método de la Secante [1]
- Método Regula Falsi [1]
- Método de Newton-Raphson [1]

Mientras que para resolver el sistema de ecuaciones no lineales se utiliza el Método Newton-Raphson Multivariable [2].

Por otro lado, los métodos de solución de sistemas de ecuaciones a utilizar son los siguientes:

- Métodos iterativos: Gauss-Jacobi [3], Gauss-Seidel [3]
- Métodos directos: LU, Cholesky, QR, LSQR y LSQR-Disperso

Para llevar a cabo la implementación de los métodos y realizar el análisis de los resultados obtenidos es necesario comprender algunos conceptos que serán explicados a continuación.

2.1. Error

El error que se aborda en el desarrolo es aquel que se asocia a la exactitud de los métodos y se relaciona con los algoritmos, donde las técnicas implementadas tienen sus propios errores.

2.2. Tolerancia

Es el valor de error que se desea obtener al usar un algoritmo numérico. Para implementar los métodos de resolución de ecuaciones se utiliza una tolerancia de 1×10^{-14} mientras que para los métodos de resolución de sistemas de ecuaciones se utiliza una tolerancia de 1×10^{-8} debido a que se trabaja con matrices y si la tolerancia fuera muy pequeña el ordenador tarda demasiado en realizar los cálculos, en especial con la matriz de dimensión 4225.

2.3. Costo temporal

Representa el número de operaciones aritméticas que se realizan. Se considera como una operación: una suma, una resta, una mutliplicación, una división o la evaluación de funciones en algún valor determinado.

2.4. Costo espacial

Corresponde al tiempo que tarda en ejecutarse el algoritmo hasta llegar a un resultado final. En la implementación se trabaja con la funciones *tic* y *toc* de Matlab para obtener el tiempo transcurrido, medido en segundos.

2.5. Especificaciones técnicas

No está demás mencionar las características técnicas que posee la máquina donde se desarrolla y ejecutan los métodos, puesto que esto puede influir en los resultados finales.

- Sistema operativo: Windows 10 Home 64 bits
- Procesador: Intel(R) Core(TM) i5-9300H CPU @ 2.40GHz (8 CPUs)
- Memoria RAM: 8192 MB

2.6. Ecuaciones

Cabe destacar que las ecuaciones no lineales a resolver son las siguientes:

$$f_1(x) : x^2 - 42$$
$$f_2(x) : x^3 - 4x^2 + 5sin(2\pi x)$$

Mientras que el sistema de ecuaciones a resolver con el método de Newton-Raphson Multivariable es $F(x)$:

$$x_1^2 + x_2 - 37 = 0$$
$$x_1 - x_2^2 - 5 = 0$$
$$x_1 + x_2 + x_3 - 3 = 0$$
$$X(0) = (1, 1, 1)^T$$

3. Resultados

3.1. Resolución de ecuaciones no lineales

Primero se resuelven las ecuaciones no lineales para las cuales se obtienen los errores mínimos representados en su forma normalizada en 4 dígitos significativos, como se puede apreciar en la Tabla 1.

Método	**Error f1**	**Error f2**
Bisección	0	$0{,}4899 \times 10^{-14}$
Secante	0	$0{,}4441 \times 10^{-15}$
Regula Falsi	$0{,}7816 \times 10^{-12}$	$0{,}1733 \times 10^{-9}$
Newton R.	0	$0{,}1332 \times 10^{-14}$

Tabla 1: Comparación de errores mínimos para f_1 y f_2

Utilizando las fórmulas vistas en clases, se obtiene el error 'a priori' de cada método, tanto para f_1 como f_2 y se considera como error 'a posteriori' aquel obtenido después de resolver el sistema con cada método, los cuales se muestran en la Tabla 2.

Método	**E. a priori**	**E. posteriori**
Bisección	$2{,}747 \times 10^{-15}$	0
Secante	$5{,}092 \times 10^{-17}$	0
Regula Falsi	$8{,}633 \times 10^{-104}$	$0{,}7816 \times 10^{-12}$
Newton R.	$3{,}589 \times 10^{-14}$	0

Tabla 2: Error a priori y posteriori para f_1

Método	**E. a priori**	**E. posteriori**
Bisección	$5{,}495 \times 10^{-15}$	$0{,}4899 \times 10^{-14}$
Secante	$1{,}193 \times 10^{-18}$	$0{,}4441 \times 10^{-15}$
Regula Falsi	0	$0{,}1733 \times 10^{-9}$
Newton R.	$8{,}816 \times 10^{-18}$	$0{,}1332 \times 10^{-14}$

Tabla 3: Error a priori y posteriori para f_2

Luego se obtienen los costos temporales y espaciales medidos en segundos y operaciones respectivamente, para obtener el error mínimo al resolver $f1$ y $f2$, obteniendo los resultados de la Tabla 4 y 5.

Método	**Tiempo [s]**	**Espacio**
Bisección	0,011893	283
Secante	0,0074667	88
Regula Falsi	0,010526	3500
Newton R.	0,0071289	28

Tabla 4: Costos temporales y espaciales para f1

Método	Tiempo [s]	Espacio
Bisección	0,0082323	275
Secante	0,0078252	72
Regula Falsi	0,096993	138600
Newton Raphson	0,0078726	20

Tabla 5: Costos temporales y espaciales para f2

Para finalizar con las ecuaciones no lineales, se procede a resolver el sistema de ecuaciones F al cual se le obtiene el error normal, absoluto y relativo en cada iteración del método Newton-Raphson, y de esta manera obtener distintas medidas de error como se aprecia en la Figura 1.

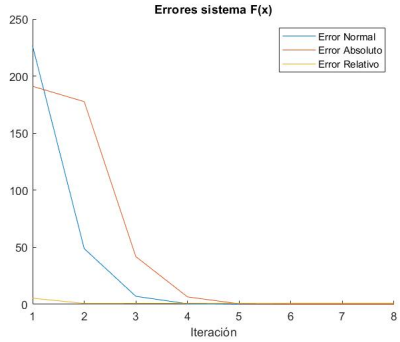

Figura 1: Errores obtenidos para F(x)

3.2. Resolución de sistemas de ecuaciones

Luego de haber implementado los algoritmos tanto iterativos como directos para la resolución de sistemas de ecuaciones, se procede a resolver los sistemas de dimensión 289x289, 1089x1089 y 4225x4225, y con el objetivo de poder identificar el método más eficaz de cada sistema, se construye una gráfica donde se expresa el error de cada uno como se puede apreciar en las Figuras 2, 3, y 4.

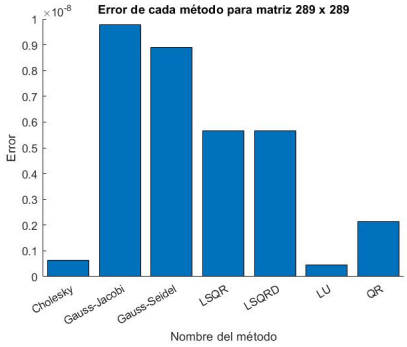

Figura 2: Errores obtenidos para matriz 289x289

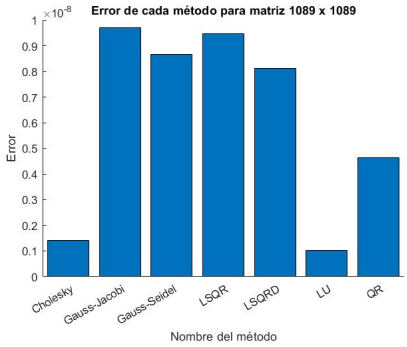

Figura 3: Errores obtenidos para matriz 1089x1089

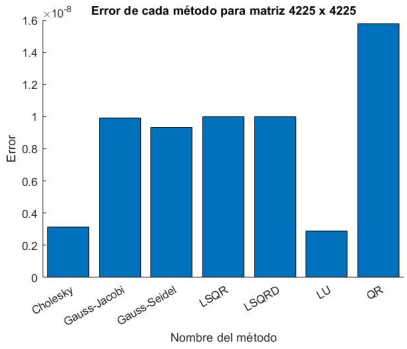

Figura 4: Errores obtenidos para matriz 4225x4225

Posteriormente, con el fin de poder identificar el método más eficiente para cada sistema, se realiza una gráfica en Matlab donde se puede apreciar el costo temporal y espacial de los métodos para cada sistema, tal como se muestra en las Figuras 5, 6 y 7.

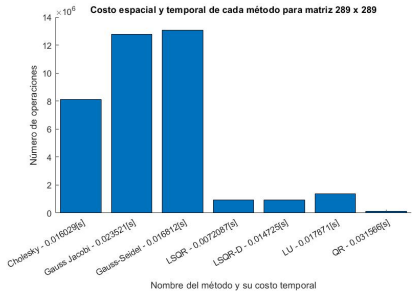

Figura 5: Costos temporales y espaciales para matriz 289x289

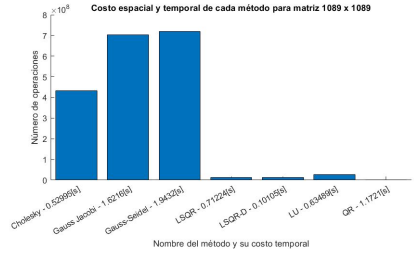

Figura 6: Costos temporales y espaciales para matriz 1089x1089

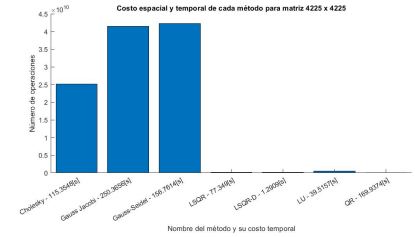

Figura 7: Costos temporales y espaciales para matriz 4225x4225

4. Discusión

A continuación se realiza un análisis de los resultados presentados anteriormente, tanto para ecuaciones no lineales como para los sistemas de ecuaciones, con el objetivo de identificar el o los métodos más eficaces y eficientes para cada caso.

4.1. Ecuaciones no lineales

Al analizar los resultados que entrega cada método para f_1 y f_2 se obtiene que el error mínimo son los que se presentan en la Tabla 1 en su representación normalizada con 4 dígitos significativos. Como se puede apreciar, el método de la Secante es el que menor error tiene para f_2, mientas que el error obtenido con el método Regula Falsi no es menor que la tolerancia definida (1×10^{-14}), y esto se debe a que el algoritmo posee una condición de parada y se detiene cuando la varianza de los últimos 100 errores es 0, lo cual indica precisión en los resultados.Con esto se puede concluir que el método más eficaz es el de la Secante.

Pero si se realiza un análisis en torno a los costos temporales y espaciales de cada método, tanto para f_1 como f_2 los métodos que menos tardan en obtener las soluciones son Newton Raphson y el método de la Secante. De hecho, son los que realizan menos operaciones, lo cual se puede observar en las Tablas 2 y 3, afirmando la eficiencia de dichos métodos.

Cabe destacar que el método de Newton Raphson calcula la derivada de la función en el punto, en cada iteración. Pero en la implementación, ésta es calculada antes de invocar al método y se entrega como parámetro al algoritmo, lo cual reduce costos y de esta manera no calcula la derivada de la función en cada iteración.

Por otro lado, analizando los errores obtenidos al resolver el sistema de ecuaciones no lineales F, es posible notar en la Figura 1 que el error normal, el cual se obtiene evaluando el punto inicial en el sistema, es el que comienza con un mayor valor pero luego disminuye más rápido que el error absoluto; mientras que el error relativo comienza con un valor pequeño. Se puede apreciar que desde la iteración 4, las tres medidas de error comienzan a converger, acercándose a 0 para obtener las raíces finales del sistema.

No está demás mencionar que el método Newton-Raphson Multivariable resuelve el sistema en un tiempo de 0.2464 segundos, con un costo temporal equivalente a 96 operaciones aritméticas, en 8 iteraciones y con un error de 1×10^{-16}, lo cual indica la eficacia y eficiencia del método.

4.2. Sistemas de ecuaciones

Como se puede apreciar en las Figuras 2, 3 y 4, el método más eficaz para los tres sistemas, es el que lo resuelve por medio de una factorización LU para la matriz A, el cual obtiene x utilizando una sustitución progresiva, e y se obtiene mediante una sustitución regresiva . Y por otro lado, Gauss-Jacobi resulta ser el menos eficaz para las matrices 289x289 y

1089x1089, mientras que para el sistema 4225x4225 resulta menos eficaz el método QR.

Por otro lado, analizando los costos temporales y espaciales de cada método, se puede apreciar cómo van aumentando los costos a medida que crece la dimensión del sistema.

Para el caso de los métodos iterativos Gauss-Jacobi y Gauss-Seidel, de acuerdo a los gráficos obtenidos se tiene que para la matriz de 289x289 y la de 4225x4225, el método menos eficiente es el método de Jacobi puesto que tarda más tiempo en entregar la solución, y los costos espaciales en comparación con Gauss-Seidel son muy cercanos. Estos métodos al ser iterativos, resulta lógico que tarden más tiempos y hagan más operaciones, ya que dependen de la tolerancia entregada (1×10^{-8}) para finalizar el algoritmo y obtener la solución, de hecho si dicha tolerancia fuera aún más pequeña, los costos aumentarían.

Tomando como referencia la matriz de 4225x4225 que es la de mayor dimensión, se obtiene que el método más eficiente es el LSQR Disperso, y esto se debe a que el método lo que hace es el mismo procedimiento que el método LSQR pero con la diferencia de que los productos matriciales son con estructuras de datos dispersas. Y como el sistema posee una cantidad de ceros, LSQR Disperso no los considera al momento de hacer los productos matriciales, lo cual disminuye notablemente los costos temporales y espaciales.

5. Conclusiones

Al dar por finalizado este laboratorio, se puede concluir que los objetivos fueron cumplidos de manera satisfactoria, pudiendo implementar en Matlab e identificar los métodos más eficaces y eficientes para cada una de las ecuaciones y sistemas de ecuaciones entregados.

Si bien, todos los métodos implementados lograron llegar a la solución para el caso de las ecuaciones no lineales, como era de esperar y de acuerdo a lo discutido en clases, el método de la Secante y de Newton-Rhapson resultaron los mas eficaces y eficientes.

En cuanto a los sistemas de ecuaciones, los método LU y Cholesky resaltan como los más eficientes. Pero viéndolo desde el punto de vista de eficiencia, los métodos directos exceptuando Cholesky, destacan como los más eficientes, mientras que los métodos iterativos Gauss-Jacobi y Gauss-Seidel resultaron los menos eficientes y menos eficaces debido a sus elevados errores y costos obtenidos de acuerdo a los gráficos.

Por otro lado, una de las complicaciones que se presentó en el desarrollo fue la resolución del sistema 4225x4225 debido a su elevada dimensión, ya que para obtener las soluciones de éste y poder graficar el costo temporal y espacial, había que esperar demasiado tiempo. Sin embargo, éste un tema que también depende de la manera en que se implementan los algoritmos, las técnicas que se utilizan y de las especificaciones que posee el ordenador donde se ejecutan, tales como la cantidad de memoria RAM y el procesador.

Si bien, Matlab ya cuenta con funciones que realizan varios de estos métodos, resulta útil conocer los algoritmos de éstos para posibles problemáticas que puedan surgir en el futuro, al desempeñarse como ingeniero informático.

Referencias

[1] Tema 2 Resolución de Ecuaciones No Lineales. (s. f.). Universidad de Granada. Recuperado 25 de mayo de 2021, de http://www.ugr.es/~mpasadas/ftp/Tema2_apuntes.pdf

[2] Newton Raphson Multivariable - danaly7. (s. f.). Danaly7. Recuperado 25 de mayo de 2021, de https://sites.google.com/site/danaly7/unidad-3/newton-raphson-modificado

[3] Métodos iterativos de Jacobi y Gauss-Seidel. (2019). Universidad Nacional Autónoma de México. Recuperado 25 de mayo de 2021, de https://www.ingenieria.unam.mx/pinilla/PE105117/pdfs/tema3/3-3_metodos_jacobi_gauss-seidel.pdf