

Actividad 3 - Método de Gauss-Seidel, Bisección, Jacobi Métodos Numéricos Ingeniería en Desarrollo de Software

Tutor: Miguel Angel Rodriguez Vega

Alumno: Jonathan Oswaldo Cardenas Garcia

Fecha: 01-noviembre-2023

Tabla De Contenido

Tabla De Contenido	2
Introducción	3
Descripción	4
Método de Jacobi	4
Método de Gauss-Seidel	4
Justificación	5
Desarrollo.....	6
Método de Bisección.....	6
Sistemas de ecuaciones con el método de Jacobi	7
Sistemas de ecuaciones con el método de Gauss-Seidel	8
Responde las siguientes preguntas:.....	9
¿Cuál es el método que te resulto más fácil de utilizar?	9
¿Cuál es el método más eficiente? ¿por qué?	9
Conclusión	10
Referencias.....	11

Introducción

En este trabajo académico se parte desde la resolución de un sistema de ecuaciones por dos métodos distintos y la creación de un programa que calcule por el método de bisección en un principio se tiene la opción de realizar estos tres problemas en Excel o en el lenguaje R, yo personalmente elegí la opción del lenguaje de programación R con su IDE RStudio, con esto se realizaron los tres pendientes primero se creó un algoritmo que solucionara ecuaciones por el método de bisección y después se crearon los argirismos para los sistemas de ecuaciones con los métodos de Jacobi y Gauss-Seidel estos métodos se usaron para resolver un sistema de ecuaciones proporcionado el cual es $(3x - y - z = 1) (-x + 3y + z = 3) (2x + y + 4z = 7)$, después se contestan un par de preguntas a manera de reflexión sobre el trabajo para posteriormente y por ultimo terminar con una conclusión sobre el trabajo académico.

Descripción

Método de Jacobi

En análisis numérico el método de Jacobi es un método iterativo, usado para resolver sistemas de ecuaciones lineales del tipo $Ax=B$. El algoritmo toma su nombre del matemático alemán Carl Gustav Jakob Jacobi. El método de Jacobi consiste en usar fórmulas como iteración de punto fijo.

La base del método consiste en construir una sucesión convergente definida iterativamente. El límite de esta sucesión es precisamente la solución del sistema. A efectos prácticos si el algoritmo se detiene después de un número finito de pasos se llega a una aproximación al valor de x de la solución del sistema.

Wikipedia.org.

Método de Gauss-Seidel

En análisis numérico el método de Gauss-Seidel es un método iterativo utilizado para resolver sistemas de ecuaciones lineales. El método se llama así en honor a los matemáticos alemanes Carl Friedrich Gauss y Philipp Ludwig von Seidel y es similar al método de Jacobi.

Aunque este método puede aplicarse a cualquier sistema de ecuaciones lineales que produzca una matriz (cuadrada, naturalmente pues para que exista solución única, el sistema debe tener tantas ecuaciones como incógnitas) de coeficientes con los elementos de su diagonal no-nulos, la convergencia del método solo se garantiza si la matriz es diagonalmente dominante o si es simétrica y, a la vez, definida positiva.

Wikipedia.org.

Justificación

En este trabajo en partículas se tenían las dos opciones para realizarlo Excel y programando con R, yo la verdad quise intentarlo programando, no fue fácil, pregunte a amigos, trate de hacer cosas que no me salieron y no termine de entender por qué y pienso que el problema está más que nada en la lógica de programación, el cómo abordar el problema matemático, sus pasos y estructura para poder hacer un algoritmo que replique lo que podríamos hacer en un Excel o a mano, fue interesante el buscar probar código de otro lado u otra solución, parchar el código con otro código que medio entendí de alguna página y que me diera errores por todos lados, pienso que tal vez para alguien experimentado algo así es fácil pero cuando no se tiene la practica aun, hasta un mínimo error es difícil de entender pero por eso elegí R para programar y tratar de entender un poco más.

Desarrollo

Método de Bisección

Figura 1

Método de Bisección ejecutado con los parámetros presentados (0, 1, 0.000001, 100)

```

14- while (i<n) {
15-   #paso 3
16-   p= a + (b-a)/2
17-   FP= Polinomio(p)
18-   print(c(i,p))
19-
20-   #paso 4
21-   if(FP==0 | (b-a)/2 < tol){
22-     return(p)
23-   }
24-   #paso 5
25-   i=i+1
26-
27-   #paso 6
28-   if (FP*FA>0){
29-     a=p
30-     FA=FP
31-   } else { b=p }
32- }
33-
34- #paso 7
35- return(paste("El metodo fallo luego de ", n, "iteraciones"))
36- }

```

```

> Raiz_biseccion(0, 1, 0.000001, 100)
[1] 7.0000000 0.6484375
[1] 8.0000000 0.6445312
[1] 9.0000000 0.6425781
[1] 10.0000000 0.6416016
[1] 11.0000000 0.6411133
[1] 12.0000000 0.6413574
[1] 13.0000000 0.6412354
[1] 14.0000000 0.6411743
[1] 15.0000000 0.6412048
[1] 16.0000000 0.6411896
[1] 17.0000000 0.6411819
[1] 18.0000000 0.6411858
[1] 19.0000000 0.6411839
[1] 20.0000000 0.6411848
[1] 0.6411848

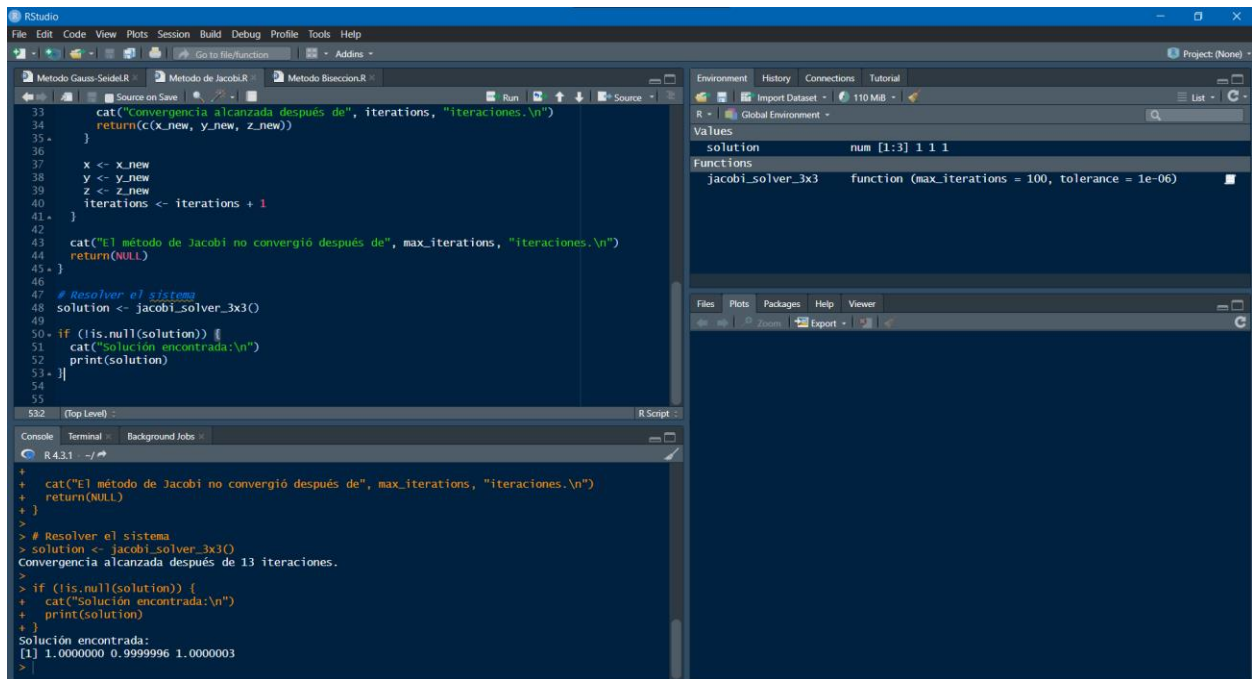
```

Nota. Método de Bisección en el lenguaje R en el IDE de RStudio, realizado con el ejemplo presentado en la clase donde se aprecia el resultado en la iteración número 20 que es 0.6411848.

Sistemas de ecuaciones con el método de Jacobi

Figura 2

Sistema de ecuaciones $(3x - y - z = 1)$ $(-x + 3y + z = 3)$ $(2x + y + 4z = 7)$ con el método de Jacobi



```

33 cat("Convergencia alcanzada después de", iterations, "iteraciones.\n")
34 return(c(x_new, y_new, z_new))
35 }
36
37 x <- x_new
38 y <- y_new
39 z <- z_new
40 iterations <- iterations + 1
41 }
42
43 cat("El método de Jacobi no convergió después de", max_iterations, "iteraciones.\n")
44 return(NULL)
45 }
46
47 # Resolver el sistema
48 solution <- jacob_i_solver_3x3()
49
50 if (!is.null(solution)) {
51   cat("Solución encontrada:\n")
52   print(solution)
53 }
54
55
532 (Top Level) - R Script

```

```

R 4.3.1 ~ /
+ cat("El método de Jacobi no convergió después de", max_iterations, "iteraciones.\n")
+ return(NULL)
+
+
>
> # Resolver el sistema
> solution <- jacob_i_solver_3x3()
Convergencia alcanzada después de 13 iteraciones.
>
> if (!is.null(solution)) {
+   cat("Solución encontrada:\n")
+   print(solution)
+ }
+
Solución encontrada:
[1] 1.0000000 0.9999996 1.0000003
>

```

Environment: Global Environment (110 MB)

Values:

name	value
solution	num [1:3] 1 1 1

Functions:

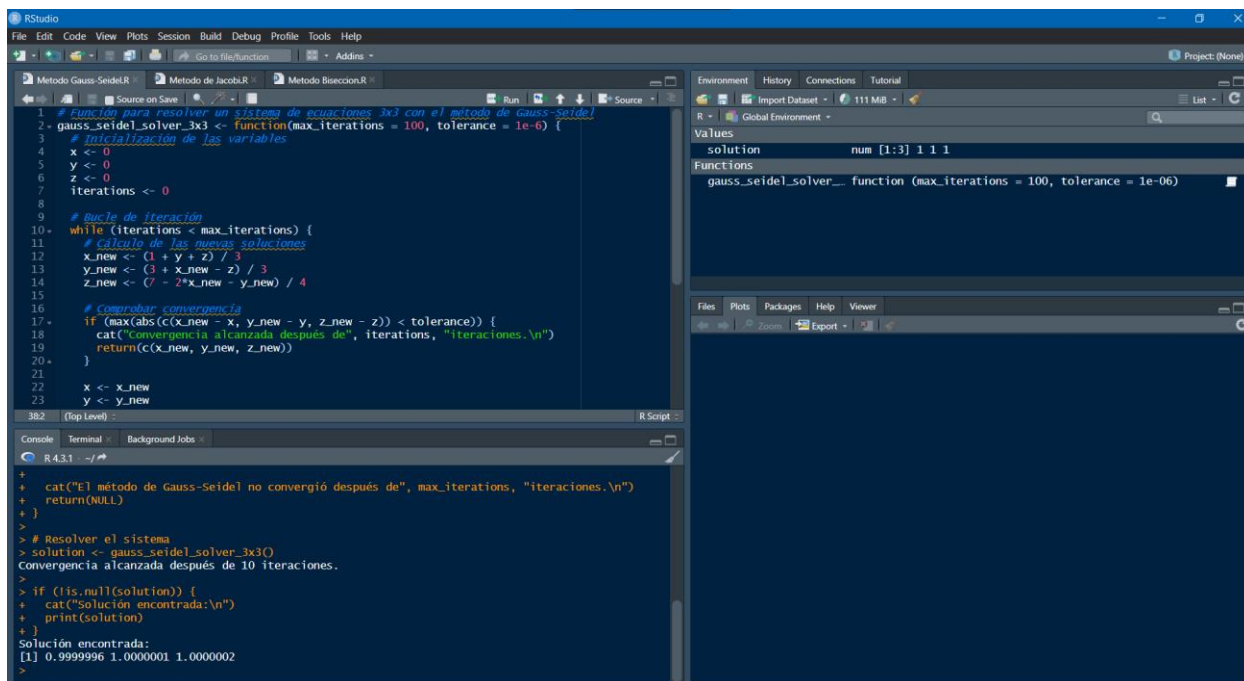
name	value
jacob_i_solver_3x3	function (max_iterations = 100, tolerance = 1e-06)

Nota. Sistema de ecuaciones realizado en el lenguaje R en su IDE RStudio mediante el método de Jacobi, para esto use condicionales (if) y un bucle (while) para poder resolver el sistema de ecuaciones, tuve un problema no supe hacer que el programa tomara las tres ecuaciones y realizara la primera paste que se mencionó en clase con el x_1 , x_2 , x_3 de las variables para sacar las nuevas ecuaciones así que las saque a mano y el programa toma como base esas tres nuevas ecuaciones del sistema que son $eq1 = (1 + y + z) / 3$ $eq2 = (3 + x - z) / 3$ y la $eq3 = (7 - 2x - y) / 4$.

Sistemas de ecuaciones con el método de Gauss-Seidel

Figura 3

Sistema de ecuaciones $(3x - y - z = 1)$ $(-x + 3y + z = 3)$ $(2x + y + 4z = 7)$ con el método de Gauss-Seidel



```

1 # Función para resolver un sistema de ecuaciones 3x3 con el método de Gauss-Seidel
2 gauss_seidel_solver_3x3 <- function(max_iterations = 100, tolerance = 1e-6) {
3   # Inicialización de las variables
4   x <- 0
5   y <- 0
6   z <- 0
7   iterations <- 0
8
9   # Bucle de iteración
10  while (iterations < max_iterations) {
11    # Cálculo de las nuevas soluciones
12    x_new <- (1 + y + z) / 3
13    y_new <- (3 + x_new - z) / 3
14    z_new <- (7 - 2*x_new - y_new) / 4
15
16    # Comprobar convergencia
17    if (max(abs(c(x_new - x, y_new - y, z_new - z)), tolerance)) {
18      cat("Convergencia alcanzada después de", iterations, "iteraciones.\n")
19      return(c(x_new, y_new, z_new))
20    }
21
22    x <- x_new
23    y <- y_new
24  }
25
26  cat("El método de Gauss-Seidel no convergió después de", max_iterations, "iteraciones.\n")
27  return(NULL)
28 }
29
30 # Resolver el sistema
31 solution <- gauss_seidel_solver_3x3()
32 cat("Convergencia alcanzada después de 10 iteraciones.\n")
33
34 if (!is.null(solution)) {
35   cat("Solución encontrada:\n")
36   print(solution)
37 }
38
39 [1] 0.9999999 1.0000001 1.0000002

```

Nota. Sistema de ecuaciones realizado en el lenguaje R en su IDE RStudio mediante el método de Gauss-Seidel, para esto use condicionales (if) y un bucle (while) para poder resolver el sistema de ecuaciones, al tener los resultados me percate que ambos resultados de los métodos coinciden pero que efectivamente este método tiene menos iteraciones, mientras que el de Jacobi tomo 13 iteraciones este solo tomo 10 para obtener el mismo resultado.

Responde las siguientes preguntas:

¿Cuál es el método que te resulto más fácil de utilizar?

Para mí el método de Jacobi porque no tiene tantas restricciones o cosas a considerar que se tienen que recordar para usarse dependiendo del caso, pienso que con este método se puede resolver de una manera un poco más fácil, aunque te tome más iteraciones pienso que vale más la pena.

¿Cuál es el método más eficiente? ¿por qué?

Eficiente como tal si nos vamos a que tomas menos iteraciones por ejemplo es el método de Gauss-Seidel en este trabajo se vio que el mismo problema con Jacobi que tomo 13 iteraciones y con Gauss-Seidel solo 10, puede que no sean muchas pero pienso que en sistemas más complejos donde son algo como 300 iteraciones unas 250 si sería un gran cambio, eso en cuanto a iteraciones se refiere pero en cuanto a complejidad del uso del método yo optaría por Jacobi, sé que no doy una respuesta tan concreta pero lo resumiré a mi opción, si ocupas enseñarle a alguien un método Jacobi es más eficiente porque se aprende antes pero si tienes todo automatizado y solo pondrás el sistema pues Gauss-Seidel tendrá menos iteraciones.

Conclusión

La verdad estas actividades me gustaron porque por un lado tenia lo normal de matemáticas que a todos nos pueden gustar más o menos por lo difícil, tedioso o cansado que se pueden hacer en algunos momentos pero por el otro lado me gusto usar el lenguaje R aunque no tenga el conocimiento para hacer cosas muy complejas el resolver un paso de la ecuación ya la mente pensaba, esto ya no lo are a mano o el Excel y si termino de programar un método completo ya solo bastara con modificar los parámetros y me generara las respuestas, ese podría ser sentimiento de progreso ya sea por terminar o evitar hacer lo mismo en matemáticas muchas veces, fue lo que termino por encender esa curiosidad de ver cómo abordar el problema y encontrar una solución, claro tal vez no de la forma más bonita o la más indicada y en ocasiones ni siquiera con la implementación que yo quería porque no sabía cómo hacerlo pero el que una actividad o materia o maestro accione esa curiosidad intelectual es muy interesante y quiero pensar que un buen comienzo.

Referencias

Liga al repositorio de GitHub

<https://github.com/CardinalSG/Metodos-Numericos.git>

(S/f). Wikipedia.org. Recuperado el 1 de noviembre de 2023, de

https://es.wikipedia.org/wiki/Método_de_Jacobi

(S/f). Wikipedia.org. Recuperado el 1 de noviembre de 2023, de

https://es.wikipedia.org/wiki/Método_de_Gauss-Seidel