Universidad Corporativa Coppel | Programas Académicos

Ingeniería en Desarrollo de Software

Actividad | 3 | Bisección

Métodos Numéricos

Ingeniería en Desarrollo de Software

TUTOR: Miguel Angel Rodríguez Vega

ALUMNO: Karol Ochoa Beltran

FECHA: 16 de noviembre 2025

Índice

Introducción

Existen diferentes métodos de aproximación, los cuales nos permiten llegar a la solución o respuesta más cercana de una función. Anteriormente hemos trabajado con los métodos de Newton-Raphson y de la secante, los cuales también sirven para lo mencionado anteriormente, pero en esta ocasión nos enfocaremos en los siguientes métodos:

Método de Bisección: Consiste en dividir el intervalo en dos mitades, cada iteración verificará el valor que tiene la función en un punto medio, este proceso se realiza en repetidas ocasiones, hasta que la iteración se anula dando como resultado la raíz. Su fórmula es $n \leq (b-a)/2^n$. Siendo "n" el numero de iteraciones, "a" y "b" los valores iniciales, estos últimos se definen otorgándole un valor a la variable "x" con el fin de obtener dos resultados con signos opuestos, es decir, uno negativo y otro positivo.

Metodo de Jacobi: Sirve para ecuaciones lineales del tipo $Ax = b$, esta funciona calculando el valor de la siguiente iteración a partir de la anterior, al igual que con el método anterior el procedimiento se realiza hasta que converge en la raíz, para que esto suceda es necesario que el valor del coeficiente perteneciente a la diagonal principal ubicada en cada una de las filas sea mayor que la suma de los valores del resto de coeficientes que se encuentren en la misma fila.

Metodo Gauss – Seidel: Este método se aplica para matrices cuadradas cuyos coeficientes sean mayores en la diagonal, para encontrar la raíz con este método se despejan las incógnitas de la ecuación y posteriormente se van cambiando por los valores más recientes que se irán actualizando con cada iteración.

Descripción

En la presente actividad implementaremos tanto el método de Jacobi como el de

Gauss-Seidel para resolver el siguiente sistema de ecuaciones: $\begin{cases} 3x & -y & -z = 1 \\ -x & +3y & +z = 3 \\ 2x & +y & +4z = 7 \end{cases}$ .Ambos métodos se realizarán en la aplicación de RStudio, la cual instalamos como primera actividad de la materia y posteriormente haremos un análisis de resultados para definir cual de los dos métodos utilizados es el que nos pareció más eficiente y sencillo para completar la actividad, en ello hay que tomar en cuenta la aproximación y la cantidad de iteraciones que resulta de cada uno.

En primera instancia, para utilizar cualquiera de estos dos métodos debemos comprobar que el sistema de ecuaciones con el que estemos trabajando, el caso del planteado en el problema es el siguiente:

$|a_{11}| > |a_{12}| + |a_{13}|$ $\quad\quad$ $|3| > |1| + |1|$

$|a_{22}| > |a_{21}| + |a_{23}|$ $\quad\quad$ $|3| > |1| + |1|$

$|a_{33}| > |a_{31}| + |a_{32}|$ $\quad\quad$ $|4| > |2| + |1|$

Por lo que podemos concluir que sí podemos aplicar ambos métodos en este sistema, a partir de aquí desarrollaremos el código en RStudio para aplicarlos y obtener el resultado aproximado a la raíz.

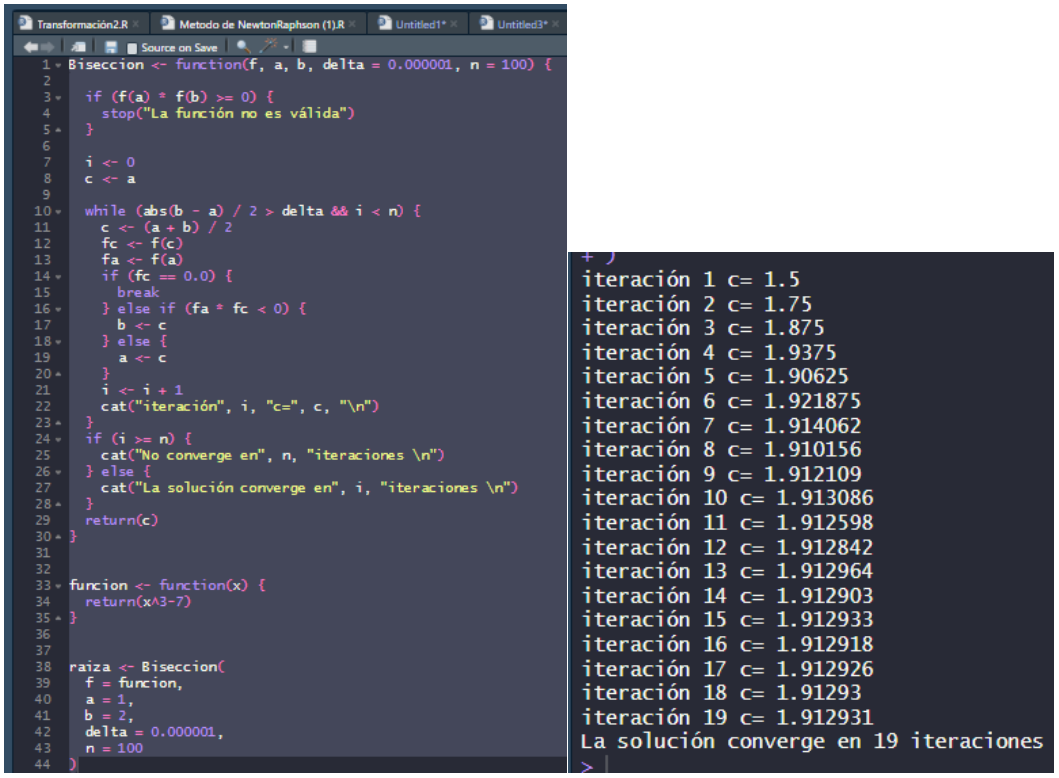
Justificación

Anteriormente comentábamos que, aunque los métodos de Gauss-Seidel, Bisección y Jacobi tienen en común la finalidad de llegar a un resultado aproximado para un problema que no tiene como tal una resolución. Lo mencionado anteriormente puede llegar a ser útil en el ámbito laboral principalmente para realizar estimados de diferentes aspectos tanto económicos como estimaciones en el mercado, ventas, etc. Además de lo mencionado anteriormente, dominar este tipo de métodos nos permite mejorar el pensamiento crítico y

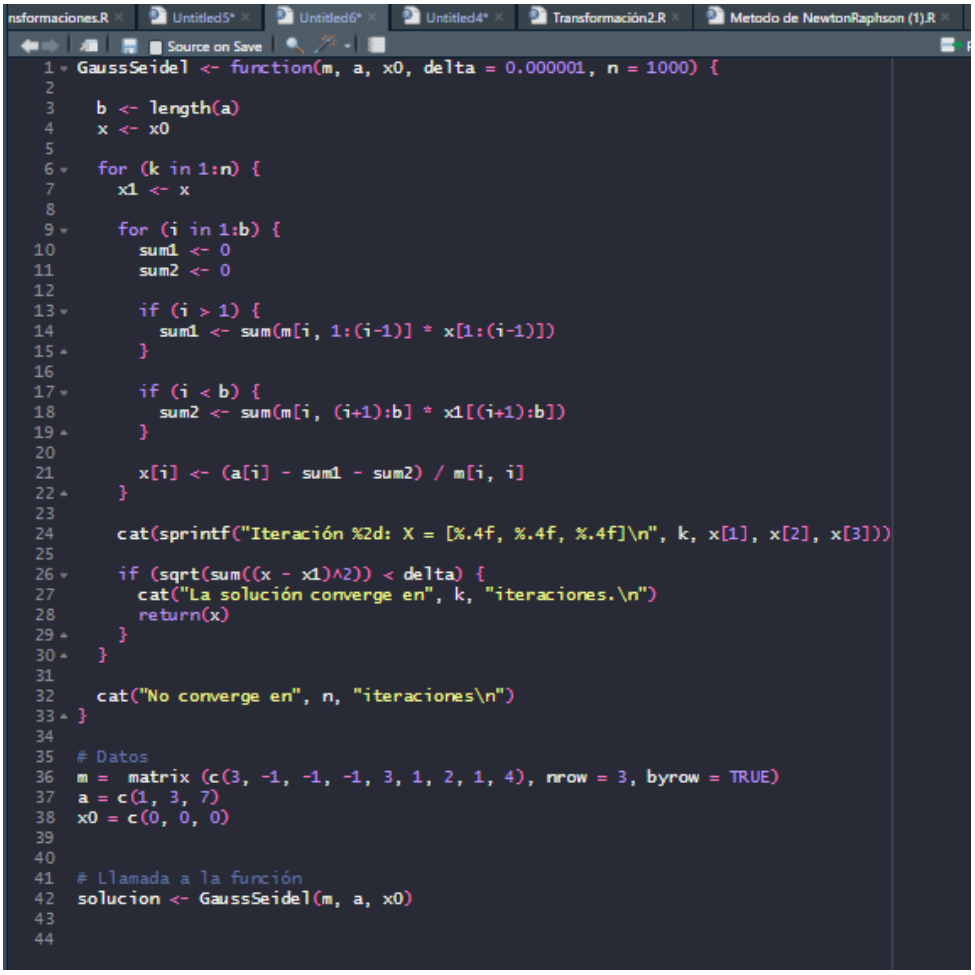
análisis de las circunstancias para una mejor toma de decisiones en base a los resultados, aunque estos no hayan sido totalmente exactos.

En base a lo mencionado anteriormente logramos definir la importancia de los métodos de aproximación para mejorar nuestra capacidad de análisis y toma de decisiones en situaciones complicadas que tengan soluciones complicadas o directamente no tengan una solución concreta para poder darle una resolución óptima para obtener los mejores resultados posibles.

Desarrollo Método de Bisección

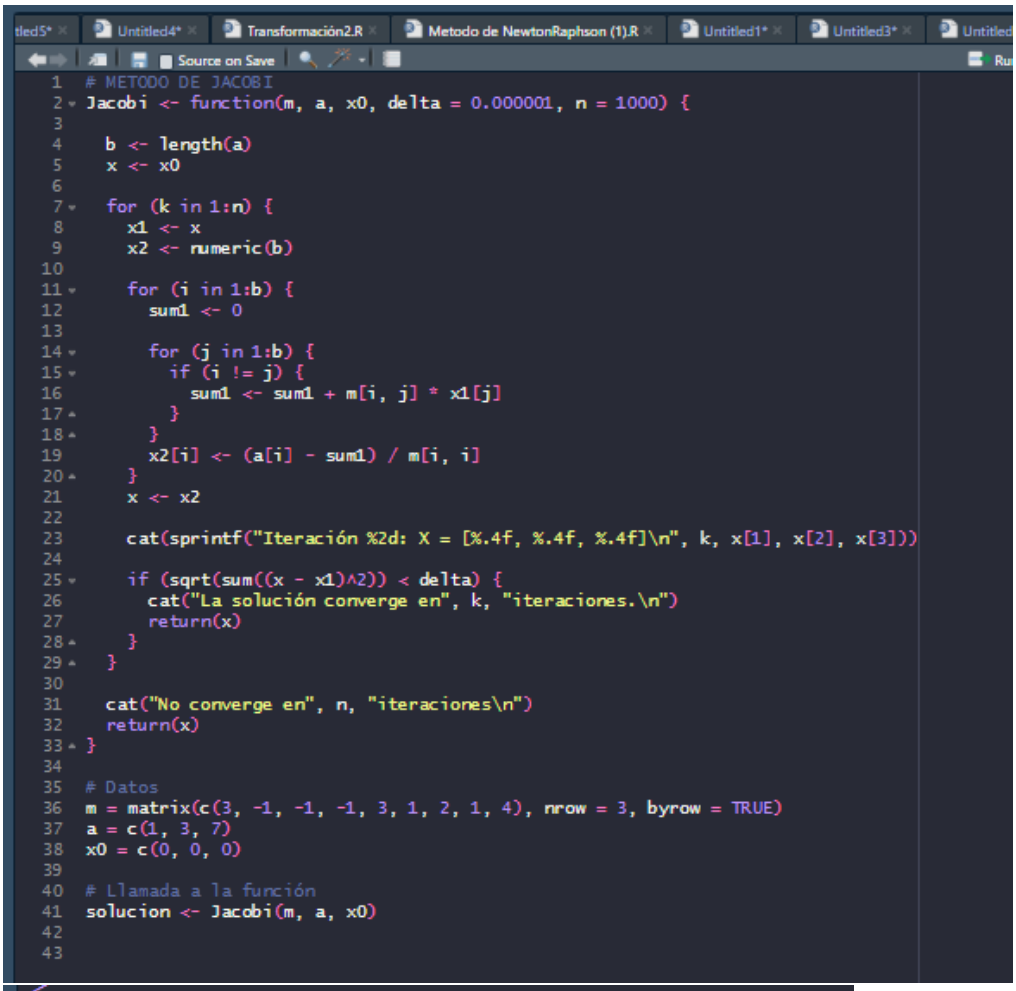


Interpretación de resultados




```r
GaussSeidel <- function(m, a, x0, delta = 0.000001, n = 1000) {

  b <- length(a)
  x <- x0

  for (k in 1:n) {
    x1 <- x

    for (i in 1:b) {
      sum1 <- 0
      sum2 <- 0

      if (i > 1) {
        sum1 <- sum(m[i, 1:(i-1)] * x[1:(i-1)])
      }

      if (i < b) {
        sum2 <- sum(m[i, (i+1):b] * x1[(i+1):b])
      }

      x[i] <- (a[i] - sum1 - sum2) / m[i, i]
    }

    cat(sprintf("Iteración %2d: X = [%.4f, %.4f, %.4f]\n", k, x[1], x[2], x[3]))

    if (sqrt(sum((x - x1)^2)) < delta) {
      cat("La solución converge en", k, "iteraciones.\n")
      return(x)
    }
  }

  cat("No converge en", n, "iteraciones\n")
}

# Datos
m =  matrix (c(3, -1, -1, -1, 3, 1, 2, 1, 4), nrow = 3, byrow = TRUE)
a = c(1, 3, 7)
x0 = c(0, 0, 0)


# Llamada a la función
solucion <- GaussSeidel(m, a, x0)
```

```
>
> # Llamada a la función
> solucion <- GaussSeidel(m, a, x0)
Iteración  1: X = [0.3333, 1.1111, 1.3056]
Iteración  2: X = [1.1389, 0.9444, 0.9444]
Iteración  3: X = [0.9630, 1.0062, 1.0170]
Iteración  4: X = [1.0077, 0.9969, 0.9969]
Iteración  5: X = [0.9979, 1.0003, 1.0009]
Iteración  6: X = [1.0004, 0.9998, 0.9998]
Iteración  7: X = [0.9999, 1.0000, 1.0001]
Iteración  8: X = [1.0000, 1.0000, 1.0000]
Iteración  9: X = [1.0000, 1.0000, 1.0000]
Iteración 10: X = [1.0000, 1.0000, 1.0000]
Iteración 11: X = [1.0000, 1.0000, 1.0000]
Iteración 12: X = [1.0000, 1.0000, 1.0000]
La solución converge en 12 iteraciones.
>
```




```
> solucion <- Jacobi(m, a, x0)
Iteración  1: X = [0.3333, 1.0000, 1.7500]
Iteración  2: X = [1.2500, 0.5278, 1.3333]
Iteración  3: X = [0.9537, 0.9722, 0.9931]
Iteración  4: X = [0.9884, 0.9869, 1.0301]
Iteración  5: X = [1.0057, 0.9861, 1.0091]
Iteración  6: X = [0.9984, 0.9989, 1.0006]
Iteración  7: X = [0.9998, 0.9992, 1.0011]
Iteración  8: X = [1.0001, 0.9996, 1.0003]
Iteración  9: X = [1.0000, 0.9999, 1.0000]
Iteración 10: X = [1.0000, 1.0000, 1.0000]
Iteración 11: X = [1.0000, 1.0000, 1.0000]
Iteración 12: X = [1.0000, 1.0000, 1.0000]
Iteración 13: X = [1.0000, 1.0000, 1.0000]
Iteración 14: X = [1.0000, 1.0000, 1.0000]
Iteración 15: X = [1.0000, 1.0000, 1.0000]
La solución converge en 15 iteraciones.
>
```

¿Cuál es el método que resultó más fácil de utilizar? En lo personal me pareció más sencillo el Jacobi, ya que anteriormente ya había trabajado de una manera similar para encontrar los valores de las variables en una ecuación.

¿Cuál es el método más eficiente? ¿Por qué? El método de Gauss-Seidel, esto sucede porque este método reemplaza el valor de la variable en cada iteración, a diferencia del de Jacobi que lo calcula en cada iteración.

Conclusión

Los métodos de aproximación son útiles para mejorar el pensamiento crítico y análisis en situaciones complicadas que pudieran presentarse tanto en nuestras vidas personales como en la laboral. Otra habilidad que seguimos desarrollando con esta actividad es el uso del lenguaje R en la aplicación de RStudio que nos permite resolver diferentes problemas matemáticos que pueden para obtener datos que pueden ser útiles principalmente en la vida laboral, ya sea para generar estimaciones de mercado, proyecciones de ventas en diferentes temporadas del año, etc. Además, al comprender cómo funcionan estos también podemos interpretar sus resultados y a partir de ahí idear diferentes maneras de optimizar los procesos ya existentes e incluso idear nuevos con el objetivo de reducir costos, pero a su vez mejorar y aumentar las ganancias generadas por la empresa. Por otro lado, en la vida cotidiana podría parecer algo no muy útil más allá de lo mencionado acerca del pensamiento crítico y análisis que nos pueden ayudar con problemas de la vida cotidiana.

Referencias

Bisection Method Definition, Steps & Solved Examples. (s. f.). Testbook.https://testbook.com/maths/bisection-method#:~:text=Bisection%20Method%20Algorithm&text=Consider%20a%20continuous%20function%20f,the%20root%20of%20the%20function.

Método de Jacoby. (s. f.). Procesos Numericos. https://procesosnumericos2015.weebly.com/meacutetodo-de-jacoby.html

Método de Jacobi. (s. f.). ScienceDirect. Recuperado 9 de noviembre de 2025, de https://www.sciencedirect.com/topics/mathematics/jacobi-method

Admin. (2022a, mayo 2). Iterative Methods Jacobi and Gauss-Seidel. BYJUS. https://byjus.com/maths/iterative-methods-gauss-seidel-and-jacobi/

De Julia Orduno, V. T. L. E. (2017, 19 marzo). Método de Jacobi y método de Gauss-Seidel. Métodos Numéricos: Portafolio. https://metodosjl.wordpress.com/2017/03/16/metodo-de-jacobi-y-metodo-de-gauss-seidel/

Gauss-Seidel. (s. f.). Procesos Numericos. https://procesosnumericos2015.weebly.com/gauss-seidel.html

De Jorgeyfloreth, L. T. L. E. (2017, 10 mayo). Método de Jacobi y método de Gauss-Seidel. Métodos Numéricos. https://jorgeyfloreth.wordpress.com/2017/03/14/metodo-de-jacobi-y-metodo-de-gauss-seidel/6

Gauss Seidel | metodosnumericos. (s. f.). Metodosnumericos. https://itcomitanisc.wixsite.com/metodosnumericos/gauss-seidel

Jmanuelcaste. (2014, 5 mayo). MÉTODO DE GAUSS SEIDEL. Análisis Número. https://esimecuanalisisnumerico.wordpress.com/2014/05/05/metodo-de-gauss-

seidel/

Link GitHub: