Universidad Corporativa Coppel | Programas Académicos

Ingeniería en Desarrollo de Software

Actividad | #3 | Métodos numéricos

Ingeniería en Desarrollo de Software

academiaglobal

TUTOR: Miguel Angel Rodríguez Vega

ALUMNO: Edgar Enrique Cuamea Ochoa

FECHA: 17 de marzo del 2024

Contenido

Introducción.

Utilizaremos Rstudio para la realización de distintos algoritmos para poder solucionar algunos problemas matemáticos utilizando operaciones menos complejas, por lo que utilizaremos este programa para realizar diferentes funciones tales como poder realizar representaciones de cantidades y variables, guardar valores fraccionados y realizar operaciones básicas como divisiones o multiplicaciones y obtener un resultado, así como poder modificar el cómo es que nos puede entregar el resultado, esto con el fin de poder utilizar los métodos numéricos para poder realizar aproximaciones de operaciones matemáticas complejas para poder solucionar un problema complejo de una forma más fácil utilizando este programa o por lo que utilizaremos los métodos de bisección así como la resolución de ecuaciones utilizando el método de jacobi y gauss-seidel así poder comprobar los resultados de las ecuaciones, por lo que debemos determinar cuál fue el método mas eficiente de estos y cual fue el más fácil de usar así comprobaremos cual de las opciones es mejor para utilizar.

Descripcion.

Tendremos en cuenta los diferentes métodos numéricos para poder encontrar el valor aproximado de una incógnita de un problema matemático complejo por lo que existen diferentes métodos que nos ayudaran a encontrar el valor aproximado, a través de estos métodos podremos utilizar diferentes formas de encontrar la incógnita utilizando diferentes funciones como tangentes o derivar funciones para optimizar el algoritmo además de tener distintos métodos con sus respectivas dificultades ya que existen métodos más simples pero con más pasos y métodos con menos pasos para encontrar el resultado aproximado de la forma más rápida posible sin tener que hacer tantas interacciones así como poder realizar operaciones más sencillas que nos tomaran más interacciones para obtener el mismo resultado por lo que usaremos el programa de Rstudio para la realización de la actividad por lo que podremos utilizar el método de jacobi y el método de gauss-seidel para comprobar la eficacia de estos métodos en la resolución de ecuaciones.

Justificación.

Aprenderemos a utilizar Rstudio para la realización de la actividad ya que es un programa para la manipulación de datos que nos funcionara para la realización de algoritmos complejos así como utilizar un lenguaje que está especializado en el análisis estadístico, este programa también nos proporciona la visualización de datos ya que podremos crear graficas que nos ayudaran a saber qué tipo de método podremos utilizar para la resolución del problema y obtener el resultado aproximado, este programa contiene funciones matemáticas y numéricas que incluyen cálculos, integraciones, diferenciación, interpolación, resolución de ecuaciones diferenciales, entre otras, podremos crear simulaciones para así poder realizar análisis de errores, además cuenta con paquetes especializados para el uso de métodos numéricos para el análisis matemático para ecuaciones diferenciales y funciones por lo que lo hace un programa practico para la resolución de problemas matemáticos complejos con el uso de funciones e interacciones repetitivas para llegar al resultado aproximado del problema.

Desarrollo.

Método de bisección,

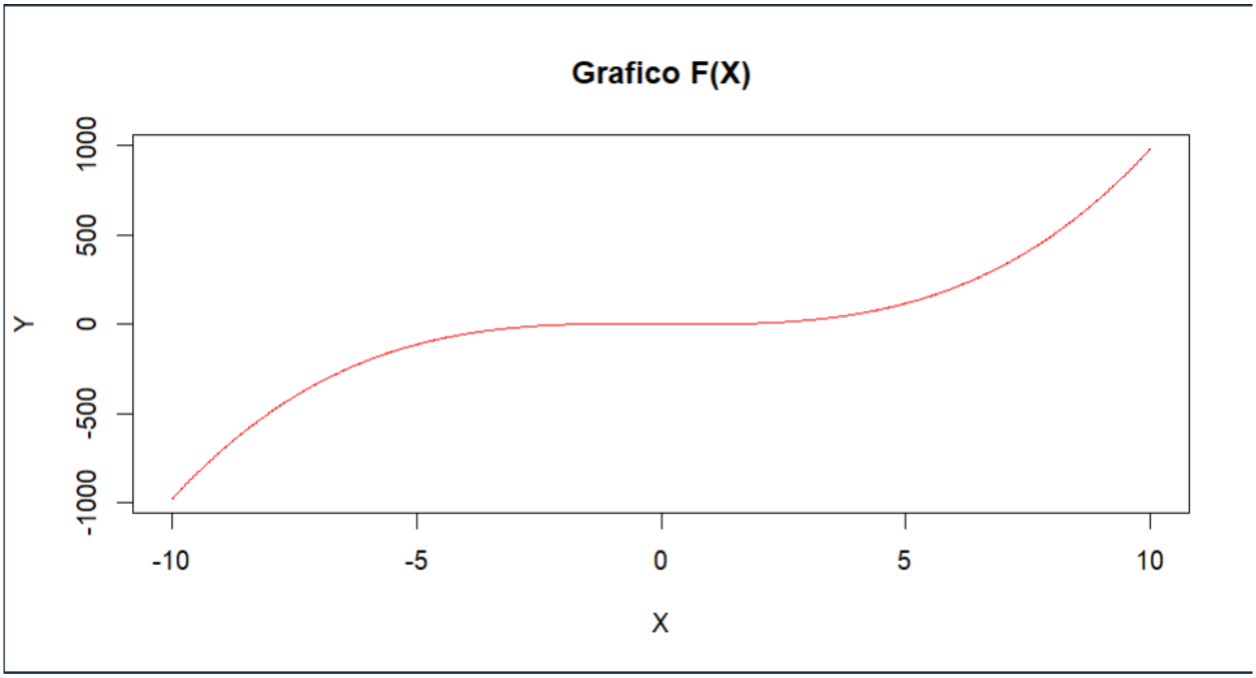
Realizaremos el código en RStudio programando así un método que realice los cálculos de manera automática por lo que es necesario saber como se realiza el método de bisección, ya que este es utilizado para encontrar raíces de funciones continuas en busca de la incógnita X donde la función se iguala a cero f(x)=0 y al ser un método iterativo donde calcula el punto medio del intervalo y verificar si el resultado es mayor al punto medio o menor a este para saber en qué sección se encuentra el resultado por lo que así completamos una interacción ya que se realizan múltiples interacciones hasta encontrar el resultado aproximado a la raíz de la función.

Necesitamos una función para poder realizar las interacciones, así como definir variables por lo que realizaremos la siguiente función $f(x)=x^3 -2x +1$

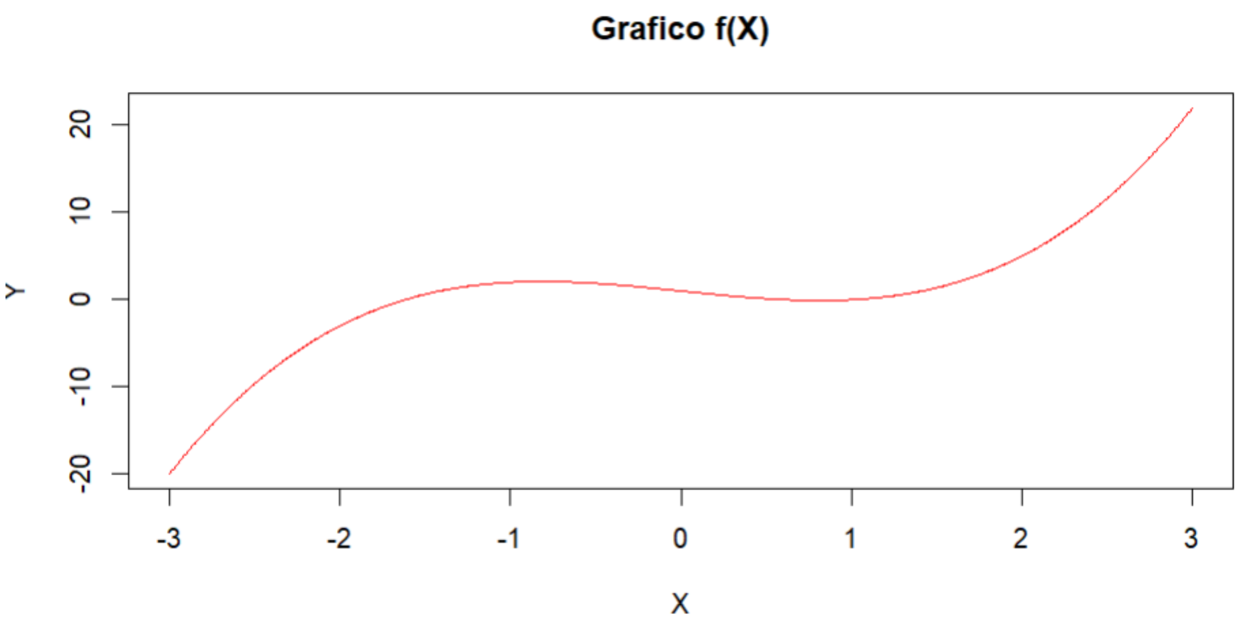
En este caso realizamos la gráfica en RStudio para verificar que la función tiene solución o no por lo que ejecutamos el siguiente código donde ingresamos la función en la variable f y representamos f en una grafica

```
> f =function(x) x^3 -2*x +1
> plot(f,-10,10,
+       lwd=1,
+       main= "Grafico F(X)",
+       col="red",
+       xlab="X",
+       ylab="Y",
+       axes=TRUE,
+       n=1000)
```

Y revisamos que la funcion tiene solucion



Por lo que reducimos valores de -3, 3 para aproximarnos al resultado



En esta grafica podemos ver que el cambio de signo sucede en 3 ocasiones por lo que esta ecuación tiene 3 soluciones aproximadamente en -1, 0 y 1 del eje X por lo que seleccionaremos

-1 y 1 para la resolución del algoritmo por lo que ingresamos la función

```
1  ecuacion =function(x) x^3 -2*x +1
2
3  raiz_Bi= function(a,b,TOL,n){
4   i=1
5   fa=ecuacion(a)
6   while (i<=n){
7     p=(a+b)/2
8     fp=ecuacion(p)
9     if (fp==0 | (b-a)/2 < TOL){
10      return(p)
11    }
12      i= i+1
13      if (fp*fa>0){
14        a=p
15        fa=fp
16      } else {
17        b=p
18      }
19  }
20 }
```

En la linea 1 definimos la ecuacion de $f(x)=x^3 -2x +1$ en una variable de función llamada ecuación y pasamos a escribir nuestro algoritmo

En la línea 3 definimos una variable llamada raíz_Bi e ingresamos en ella 4 valores distintos, es importante tener esta variable para ingresar los valores en un solo comando ya que el no tener esta variable hace más complejo el proceso de asignar valores así como el algoritmo puede fallar, por lo que ingresamos el punto a, el punto b, la tolerancia o el error absoluto y el número de interacciones máxima en n todas estas como variables sin ingresar números en ellas, esto hará que al ingresar estos valores ejecutara los siguientes comandos, en la línea 4 una variable i donde guardara las interacciones, calculamos fa donde guardaremos la ecuación con el valor actual de a, inicializamos un bucle while donde se ejecutara siempre que el valor de i sea menor o igual a n, ya que establecemos un número máximo de interacciones en n, en nuestra línea 7 calculamos el punto medio donde este será el valor a mas b sobre 2 y lo asignamos a la variable p y calculamos la ecuación con el valor actual de p en la variable fp.

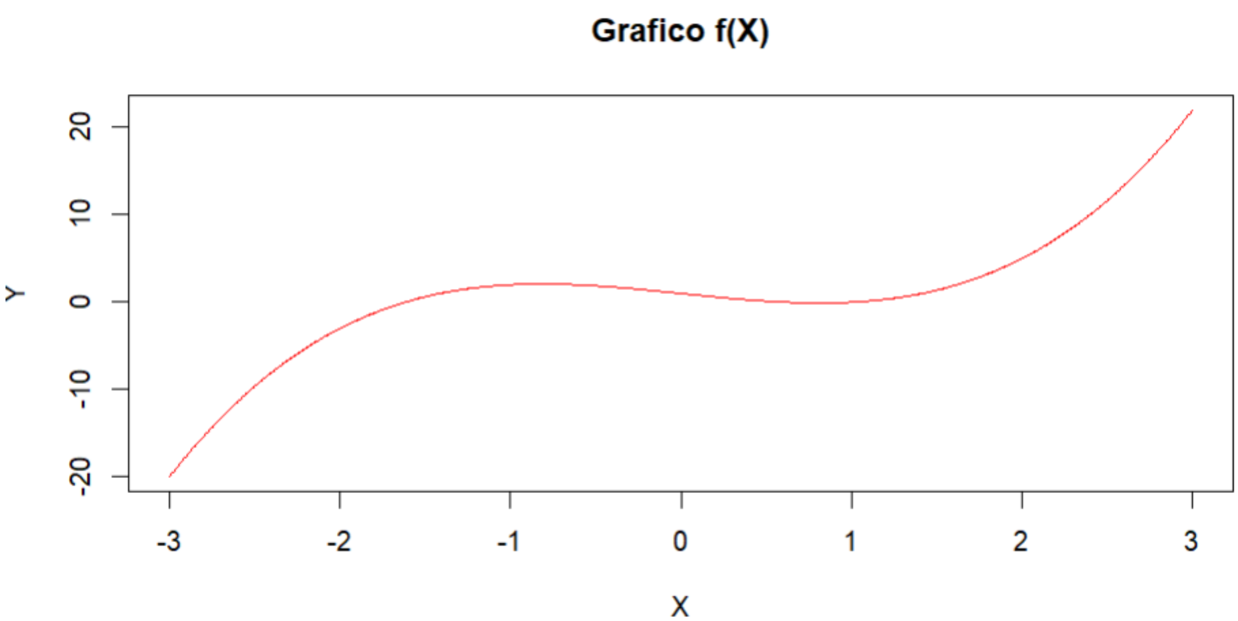
En este punto calcularemos el valor de el punto medio para comprobar si el resultado se encuentra entre el punto medio y a o el punto medio b. agregamos una condición donde si el valor de fp es igual a cero o si la resta entre los valores de b menos a divididos entre dos es menor a la tolerancia asignada, este mostrará el valor de p ya que esa será la solución, en este caso r studio utiliza el signo (|) para comparar resultados dentro de la condicional, en caso de que no se cumpla ninguna de las condiciones, ejecutara lo siguiente

```
    }
      i= i+1
      if (fp*fa>0){
        a=p
        fa=fp
      } else {
        b=p
      }
  }
}
```

Actualizara la variable i sumandole uno por cada interaccion por lo que agregamos una condicional donde si la funcion de el punto medio por la funcion del punto inicial a es mayor a 0 entonces el valor de a pasa a ser actualizado a el valor de p y fa se actualiza con el valor de fp ya que si las funciones fp y fa tienen el mismo signo, significa que la raiz no se encuentra entre el punto medio y el punto inicial a, ya que buscamos el resultado de la raiz donde la funcion pase de ser signo negativo a signo positivo y en caso de que se cumpla esta condicion entonces la raiz encuentra entre a y p por lo que b pasa a ser el valor de el punto medio para calcular denuevo el algoritmo hasta encontrar la raiz con un error de 10^{-6} o al llegar al numero maximo de interacciones que definimos por lo que llamamos la funcion raiz_Bi

```
> raiz_Bi(0,1,0.000001,100)
[1] 0.6180334
> raiz_Bi(-5,0,0.0000001,100)
[1] -1.618034
> raiz_Bi(1,3,0.0000001,100)
[1] 1
> raiz_Bi(-10,10,0.0000001,100)
[1] -1.618034
>
```

En este caso seleccionamos los valores 0,1 para el primer resultado, -5,0 para el segundo y 1,3 para el tercer resultado



Coincidiendo con los valores aproximados donde la funciones cambian de signo dandonos así los resultados aproximados, ademas de comprobar que la funcion tiene 3 soluciones posibles, en este metodo tendremos que ajustar los valores a y b aproximados a la solucion de la funcion en base a la grafica ya que en el cuarto resultado elegimos los valores de -10,10 y este algoritmo encontro el valor de -1.618034 por lo que es facil llamar la funcion para cambiar los valores de la tolerancia y los valores a y b así como las interacciones ademas de tener una variable exclusiva con la funcion que queremos calcular ya que hace facil cambiar la funcion y volver a calcular el valor aproximado de la solucion o la raiz de la ecuacion. ya que anteriormente no agrege una variable que guardara el algoritmo y calculara todo desde la funcion principal lo que hizo un algoritmo largo y complejo para calcular la funcion y al cambiar los valores de a y b así como la funcion a calcular, el algoritmo fallaba ya que no encontraba los valores de las variables adecuadamente ya que estaban incluidas en el principio del algoritmo y sin tener un apartado solo para las variables ya que solo las habia agregado una por una sin llamar una variable para poderlas asignar en una sola linea y despues ejecutar el codigo y al realizarlo de esta forma este algoritmo fallaba por lo que es demasiado importante guardar nuestro algoritmo en una variable que podremos llamar así como tener la ecuacion en una variable aparte de el algoritmo para evitar problemas y errores por el simple hecho de no realizarlo correctamente, por lo que entender como es que se realiza el metodo de biseccion y trasladarlo a codigo fue una tarea compleja al olvidar poner las variables por separado ya que el metodo es sencillo y al tener la ecuacion separadas de las variables y de el algoritmo, hace el proceso de calcular diferentes ecuaciones mas facil.

Podremos ingresar los comandos cat(i, a, b, p, fp, "\n") despues de nuestra primera condicional para mostrar una tabla donde nos muestre los resultados de cada interaccion.

Método de Jacobi.

$$\begin{cases} 3x - y - z = 1 \\ -x + 3y + z = 3 \\ 2x + y + 4z = 7 \end{cases}$$

Tenemos nuesrta diagonal dominante 3x, 3y,4z ya que estos contienen los valores mas altos por lo que se realiza el despeje de x por lo que quedaria de la siguiente forma

X= -y -z +1 /3

Y= -x +z +3 /3

Z= -2x +y +7 /4

Asignamos valores iniciales = 0 y realizamos nuestro despeje en exel para cada una de las variables

	A	B	C	D	E	F	G
1	Iteracciones	X	Y	Z	Error X	Error Y	Error Z
2	0	0	0	0			0
3	1	=(-C2-D2+1)/3					1
4	2						2
5	3						3
6	4						4
7	5						5
8	6						6

Realizamos las formulas en exel para descubrir los resultados de las variables en la primera interaccion, calculamos nuestro error absoluto de X que es el resultado de X en la interaccion 1 menos X en la interaccion 0 entre X en interaccion 1 por lo que el resultado es el siguiente

E3 =ABS(B3-B2)/B3

	A	B	C	D	E	F	G
1	Iteracciones	X	Y	Z	Error X	Error Y	Error Z
2	0	0	0	0			
3	1	0.33333333	1	1.75	1	1	1
4	2	-0.5833333	1.47222222	1.83333333	-1.5714286	0.32075472	0.04545455
5	3	-0.7685185	1.80555556	2.40972222	-0.2409639	0.18461538	0.23919308
6	4	-1.0717593	2.05941358	2.58564815	-0.2829374	0.12326714	0.06803939
7	5	-1.2150206	2.2191358	2.80073302	-0.1179086	0.07197497	0.07679592
8	6	-1.3399563	2.33858453	2.91229424	-0.0932386	0.05107736	0.03830699
9	7	-1.4169596	2.41741684	3.00462427	-0.054344	0.03261014	0.03072931
10	8	-1.4740137	2.47386129	3.062834	-0.0387066	0.02281634	0.01900519
11	9	-1.5122318	2.51228257	3.10547217	-0.0252726	0.01529338	0.01373001
12	10	-1.5392516	2.53923465	3.13418652	-0.0175539	0.01061425	0.00916166
13	11	-1.5578071	2.5578127	3.15443445	-0.0119113	0.00726326	0.00641888
14	12	-1.5707491	2.57074717	3.1683567	-0.0082394	0.0050314	0.00439416
15	13	-1.5797013	2.57970192	3.17806132	-0.005667	0.00347123	0.00305363
16	14	-1.5859211	2.58592087	3.18477613	-0.0039219	0.00240493	0.00210841
17	15	-1.5902323	2.5902324	3.18944076	-0.0027111	0.00166453	0.00146252
18	16	-1.5932244	2.59322436	3.19267427	-0.001878	0.00115376	0.00101279
19	17	-1.5952995	2.59529955	3.19491828	-0.0013008	0.00079959	0.00070237
20	18	-1.5967393	2.59673928	3.19647466	-0.0009017	0.00055444	0.0004869
21	19	-1.597738	2.59773798	3.19755446	-0.0006251	0.00038445	0.0003377
22	20	-1.5984308	2.59843081	3.19830348	-0.0004334	0.00026664	0.00023419
23	21	-1.5989114	2.59891143	3.19882311	-0.0003006	0.00018493	0.00016244
24	22	-1.5992448	2.59924485	3.19918357	-0.0002085	0.00012827	0.00011267
25	23	-1.5994761	2.59947614	3.19943364	-0.0001446	8.8977E-05	7.8158E-05
26	24	-1.5996366	2.59963659	3.19960711	-0.0001003	6.1721E-05	5.4216E-05
27	25	-1.5997479	2.5997479	3.19972744	-6.958E-05	4.2815E-05	3.7609E-05
28	26	-1.5998251	2.59982511	3.19981092	-4.826E-05	2.97E-05	2.6089E-05
29	27	-1.5998787	2.59987868	3.19986884	-3.348E-05	2.0603E-05	1.8098E-05
30	28	-1.5999158	2.59991584	3.19990901	-2.323E-05	1.4292E-05	1.2555E-05
31	29	-1.5999416	2.59994162	3.19993688	-1.611E-05	9.9147E-06	8.7093E-06
32	30	-1.5999595	2.5999595	3.19995621	-1.118E-05	6.8779E-06	6.0417E-06
33	31	-1.5999719	2.5999719	3.19996962	-7.753E-06	4.7713E-06	4.1912E-06

A continuacion deslizamos hacia abajo para automatizar las operaciones y revisamos los errores absolutos de X,Y y Z para comprobar que los errores bajen en cada una de ellas o por lo menos baje en una variable por cada iteraccion para comprobar que el metodo funciona correctamente,

Podemos ver que en la interaccion 31 se cumple la condicion de un error permitido de 10^{-6} por lo que podriamos terminar el procedimiento dando como resultado X=-1.5999 Y=2.5999 Z=3.1999

89	-1.6	2.6	3.2	-4.857E-15	2.9037E-15	2.6368E-15
90	-1.6	2.6	3.2	-3.331E-15	2.0496E-15	1.6653E-15
91	-1.6	2.6	3.2	-2.22E-15	1.3664E-15	1.3878E-15
92	-1.6	2.6	3.2	-1.665E-15	1.0248E-15	8.3267E-16
93	-1.6	2.6	3.2	-1.11E-15	6.8321E-16	5.5511E-16
94	-1.6	2.6	3.2	-6.939E-16	5.1241E-16	4.1633E-16
95	-1.6	2.6	3.2	-5.551E-16	3.4161E-16	2.7756E-16
96	-1.6	2.6	3.2	-4.163E-16	1.708E-16	1.3878E-16
97	-1.6	2.6	3.2	-1.388E-16	1.708E-16	2.7756E-16
98	-1.6	2.6	3.2	-4.163E-16	0	0
99	-1.6	2.6	3.2	0	1.708E-16	1.3878E-16
100	-1.6	2.6	3.2	0	0	0

Revisando las interacciones hasta llegar el resultado, vemos que en la interaccion 100 llegamos a la solucion donde X=-1.6 Y=2.6 Z=3.2 dando como errores valores 0

Metodo de Gauss-Seidel

Utilizaremos el mismo sistema de ecuaciones para este método por lo que el error permitido será el mismo de 10^{-6}

$$\begin{cases} 3x - y - z = 1 \\ -x + 3y + z = 3 \\ 2x + y + 4z = 7 \end{cases}$$

Realizaremos el mismo proceso de encontrar la diagonal dominante 3x, 3y,4z ya que es una modificacion del metodo de jacobi por lo que realizaremos las mismas formulas para el despeje de las incognitas

X= -y -z +1 /3

Y= -x +z +3 /3

Z= -2x +y +7 /4

Lo ingresamos a exel con los valores iniciales en 0

C3 =(-B3+D2+3)/3

	A	B	C	D	E	F	G	H
1	Iteracciones	X	Y	Z	Error X	Error Y	Error Z	
2	0	0	0	0				
3	1	0.33333333	=(-B3+D2+3)/3					
4								
5								

Realizamos el mismo metodo para encontrar el valor de x en la primera interaccion y realizamos el despeje de Y solo que en vez de seleccionar x en su interaccion 0, seleccionamos x en su interaccion 1 para acelerar el proceso de encontrar la solucion y en caso de z, seleccionamos z en interaccion 0 ya que aun no calculamos su valor

D3 =(-2*B3+C3+7)/4

	A	B	C	D	E
1	Iteracciones	X	Y	Z	Error X
2	0	0	0	0	
3	1	0.33333333	0.88888889	=(-2*B3+C3+7)/4	
4					
5					

Para encontrar el valor de Z, realizamos el despeje seleccinando X y Y en las interacciones 1 ya que son las que acabamos de calcular anteriormente y vemos que los valores cambian respecto al metodo de jacobi

	A	B	C	D	
1	Iteracciones	X	Y	Z	Error
2	0	0	0	0	
3	1	0.33333333	0.88888889	1.80555556	
4					

Calculamos el error absoluto y verificamos que esten en valores 1

	A	B	C	D	E	F	G
1	Iteracciones	X	Y	Z	Error X	Error Y	Error Z
2	0	0	0	0			
3	1	0.33333333	0.88888889	1.80555556	1	1	1
4							
5							

Realizamos el mismo procedimiento que en el metodo de jacobi y revisamos que nuestros errores esten disminuyendo

A	B	C	D	E	F	G
2	-0.5648148	1.79012346	2.47993827	-1.5901639	0.50344828	0.27193528
3	-1.0900206	2.18998628	2.84250686	-0.4818311	0.18258691	0.12755241
4	-1.3441644	2.39555708	3.02097146	-0.189072	0.08581336	0.05907524
5	-1.4721762	2.49771588	3.11051706	-0.0869541	0.04090089	0.02878801
6	-1.5360776	2.5488649	3.15525505	-0.0416004	0.02006737	0.01417888
7	-1.56804	2.57443168	3.17762791	-0.0203836	0.00993104	0.00704074
8	-1.5840199	2.58721592	3.18881391	-0.0100882	0.00494131	0.00350789
9	-1.5920099	2.59360795	3.19440696	-0.0050189	0.00246453	0.00175089
10	-1.596005	2.59680398	3.19720348	-0.0025031	0.00123075	0.00087468
11	-1.5980025	2.59840199	3.19860174	-0.00125	0.000615	0.00043715
12	-1.5990012	2.59920099	3.19930087	-0.0006246	0.0003074	0.00021853
13	-1.5995006	2.5996005	3.19965044	-0.0003122	0.00015368	0.00010925
14	-1.5997503	2.59980025	3.19982522	-0.0001561	7.6833E-05	5.4623E-05
15	-1.5998752	2.59990012	3.19991261	-7.803E-05	3.8415E-05	2.7311E-05
16	-1.5999376	2.59995006	3.1999563	-3.902E-05	1.9207E-05	1.3655E-05
17	-1.5999688	2.59997503	3.19997815	-1.951E-05	9.6035E-06	6.8275E-06
18	-1.5999844	2.59998752	3.19998908	-9.754E-06	4.8017E-06	3.4137E-06

Podemos ver que en la interaccion 18 se llega a la aproximacion del resultado aproximado de la solucion con el error permitido de 10^{-6}, así que seguimos buscando la solucion y con este metodo se llega a la interaccion 54 mostrando los mismos valores que encontramos para la incognita con el metodo de jacobi

49	-1.6	2.6	3.2	-4.58E-15	2.2204E-15	1.5266E-15
50	-1.6	2.6	3.2	-2.22E-15	1.0248E-15	8.3267E-16
51	-1.6	2.6	3.2	-1.11E-15	5.1241E-16	2.7756E-16
52	-1.6	2.6	3.2	-5.551E-16	3.4161E-16	2.7756E-16
53	-1.6	2.6	3.2	-4.163E-16	1.708E-16	1.3878E-16
54	-1.6	2.6	3.2	0	0	0
55	-1.6	2.6	3.2	0	0	0

1.- ¿Cuál es el método que resultó más fácil de utilizar?

El metodo de jacobi en exel es mas facil de utilizar ya que realizamos todas las variables con valores de 0 ademas de calcular el error absoluto de una manera mas facil al poder arrastrar todo en conjunto y realizar las interacciones de manera automatica

2.- ¿Cuál es el método más eficiente?

El metodo de biseccion en rstudio

¿Por qué?

Ya que realizamos un algoritmo, solo tendremos que agregar una funcion y asignar valores de puntos a calcular, errores perminitos e interacciones maximas para el algoritmo ya que el metodo de gauss, solo es una modificacion de el metodo de jacobi por lo toma valores calculados actuales de X y Y por lo que lo hace mas eficiente que el de jacobi.

Conclusión.

Realizamos los métodos de jacobi y gauss para calcular una función lineal donde comparamos los resultados obtenidos de los dos métodos, aunque estos dos métodos funcionan relativamente igual, el método de gauss es mas eficiente al tener menos interacciones para obtener los mismos resultados comparados con el método de Jacobi, realizamos el código del método de bisección en r studio donde fue una tarea sencilla de entender su proceso y solución por lo que al trasladarlo a código fue una tarea complicada al tener que realizar el código sin que fallara ya que es necesario saber que las variables las definiremos primero antes de ejecutar el código y tener variables especiales solo para las ecuaciones a calcular ya que al no realizarlo correctamente el código falla al no encontrar las variables al momento de asignarlas, realizarlo en exel es una tarea sencilla de realizar ya que podremos realizar las ecuaciones de forma mas fácil que programarlas en r studio pero el programar un algoritmo que realice las operaciones para encontrar el resultado de las ecuaciones siempre será mas eficiente que realizarlo de forma manual al ejecutarlo mas rápido por este algoritmo.

Referencias.

Doruklar C.A. (2017, 13 julio). *Método de Bisección implementado en R* [Vídeo]. YouTube. https://www.youtube.com/watch?v=8Hnvioh74DY

Nuevo Espacio CECE (Canal Oficial 2020). (2020, 4 abril). *Análisis Numérico. Método de Bisección en R . Prof. Darío Bacchini* [Vídeo]. YouTube. https://www.youtube.com/watch?v=7OfJSQown8U

Hackemi. (s. f.-a). *GitHub - Hackemi/Metodos_numericos*. GitHub. https://github.com/Hackemi/Metodos_numericos