

Universidad Mariano Gálvez

Métodos Numéricos

Sección 2

Ciclo 2024



Proyecto Final Métodos Numéricos
(Método de eliminación Gaussiana)

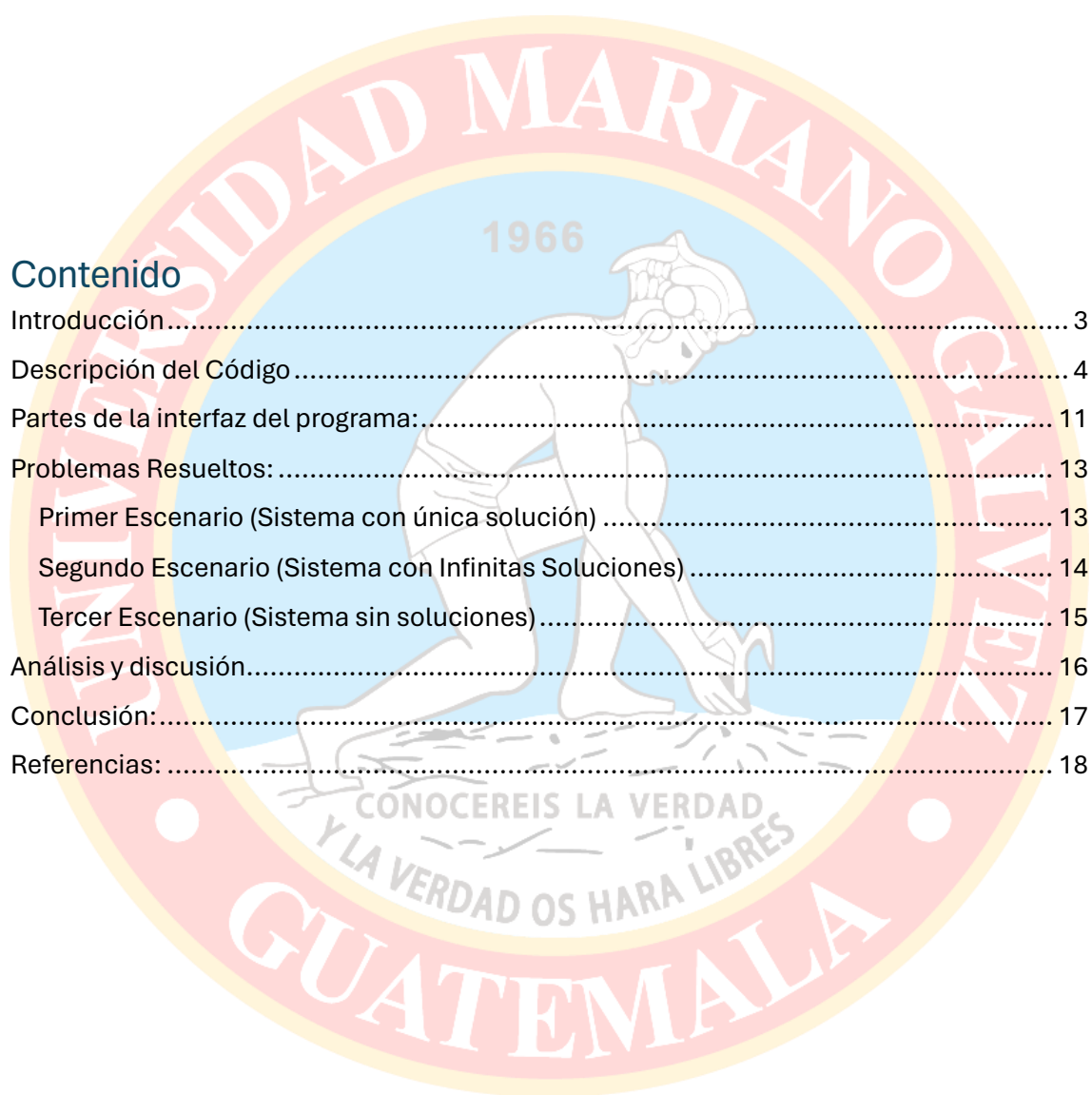
Jorge Mario Monzón González

5990-20-11568

Guatemala, junio de 2024

Contenido

Introducción.....	3
Descripción del Código	4
Partes de la interfaz del programa:.....	11
Problemas Resueltos:	13
Primer Escenario (Sistema con única solución)	13
Segundo Escenario (Sistema con Infinitas Soluciones)	14
Tercer Escenario (Sistema sin soluciones)	15
Análisis y discusión.....	16
Conclusión:.....	17
Referencias:	18



Introducción

En la búsqueda de encontrar resolver sistemas matemáticos como lo son los sistemas de ecuaciones de 3 incógnitas se han desarrollado diferentes métodos que permiten encontrar matemáticamente los valores de cada una de esas incógnitas, pero existen problemas que no pueden resolverse con valores exactos ya que por incluir factores de error en las mediciones de campo o en los instrumentos que se utilizan estos pueden no ser tan precisos, de esta forma se descubrió que si se utilizan las reglas de eliminación gaussiana se pueden resolver estos sistemas de ecuaciones de una forma en la que los valores para cada una de las incógnitas son aproximados, y además permite la creación de algoritmos que son repetitivos y evitan grandemente el error en los cálculos por factores humanos al momento de hacer las sustituciones ya que el algoritmo de eliminación gaussiana es intuitivo y lineal que permite encontrar si un sistema contiene única solución, soluciones infinitas o incluso si no tiene solución, con los avances tecnológicos hemos podido delegar a las maquinas las tareas básicas de calculo y computo, pero es la continua mejora y experiencia de los ingenieros que se han podido trasladar estos algoritmos a nivel de programación para que las maquinas ya no solo tengan la capacidad de realizar los cálculos por nosotros sino que también puedan realizar esa secuencia lineal de pasos y revolverlos de formas donde la aproximación hace que el factor de error sea muy bajo. En esta ocasión me he servido de la tecnología para realizar un algoritmo o método numérico llamado eliminación gaussiana el cual nos permitirá resolver este tipo de ecuaciones solo con digitar los valores de las ecuaciones y con solo presionar el botón de calcular podremos ver la secuencia de pasos que la maquina utilizara, el algoritmo en cada iteración y así mismo nos revelara si la ecuación tiene solución y cuales son en caso de tenerlas.

Descripción del Código

Para la aplicación del método numérico se utiliza el método de eliminación Gaussiana que toma un sistema de ecuaciones de primer grado y lo convierten en una matriz de coeficientes o también llamada matriz aumentada.

Como en el ejemplo:

$$\begin{array}{rrcr} +2X & +6Y & +1Z & = +7 \\ +1X & +2Y & -1Z & = -1 \\ +5X & +7Y & -4Z & = +9 \end{array}$$

Este sistema de ecuaciones está formado por tres ecuaciones con 3 incógnitas con el que se obtiene la matriz aumentada siguiente:

$$\left| \begin{array}{ccc|c} 2 & 6 & 1 & 7 \\ 1 & 2 & -1 & -1 \\ 5 & 7 & -4 & 9 \end{array} \right|$$

El código de programación encargado de realizar esta transformación, se apoya primero con un conjunto de validaciones que rellena con 0 los valores de las casillas que el usuario deje vacías con la finalidad de evitar errores de sistema.

Ingreso de Datos:			
X	Y	Z	=
2	6	1	7
1	2	-1	-1
5	7	-4	9

```
Seteando Ceros en casillas vacias de X
Seteando Ceros en casillas vacias de Y
Seteando Ceros en casillas vacias de Z
Seteando Ceros en casillas vacias de las igualdades
Seteando Ceros con Exito....
```

Fragmento de código:

Esta funcionalidad se realiza por cada uno de los valores ingresados por el usuario.

```
Private Sub PL_colocar_ceros()  
    Try  
        'colocando ceros a los valores de X  
        Write_log("Seteando Ceros en casillas vacias de X")  
        If txt_x1.Text = "" Then  
            txt_x1.Text = "0"  
        End If  
        If txt_x2.Text = "" Then  
            txt_x2.Text = "0"  
        End If  
        If txt_x3.Text = "" Then  
            txt_x3.Text = "0"  
        End If  
    End Try  
End Sub
```

El siguiente paso es validar que el usuario haya ingresado solo valores numéricos y no letras o caracteres especiales.

```
Validar si los valores ingresados son numericos....  
Valores de X correctos...  
Valores de Y correctos...  
Valores de Z correctos...  
Valores de las igualdades son correctos...  
Valores de los campos son numericos....
```

Fragmento de código:

Se valida cada uno de los valores ingresados para detectar si son numéricos y si no lo son se retorna el mensaje de error al usuario para que pueda rectificar la información.

```
Private Function PL_validad_campos_numericos() As Boolean  
    Try  
        Write_log("Validar si los valores ingresados son numericos....")  
        'validando si los valores de X son numericos  
        If IsNumeric(txt_x1.Text) And IsNumeric(txt_x2.Text) And IsNumeric(txt_x3.Text) Then  
            Write_log("Valores de X correctos...")  
        Else  
            Write_log("Revise los valores de X ya que no son numericos")  
            MsgBox("Revise los valores de X ya que no son numericos")  
            Return False  
        End If  
  
        'validando si los valores de Y son numericos  
        If IsNumeric(txt_y1.Text) And IsNumeric(txt_y2.Text) And IsNumeric(txt_y3.Text) Then  
            Write_log("Valores de Y correctos...")  
        Else  
            Write_log("Revise los valores de Y ya que no son numericos")  
            MsgBox("Revise los valores de Y ya que no son numericos")  
            Return False  
        End If  
    End Try  
End Function
```

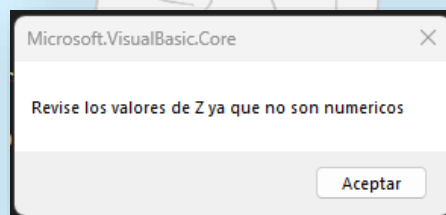
Mensaje de error:

```
Validar si los valores ingresados son numericos....  
Valores de X correctos...  
Valores de Y correctos...  
Valores de Z correctos...  
Valores de las igualdades son correctos...  
Valores de los campos son numericos....
```

Se ingresa un valor incorrecto en la columna Z3

X	Y	Z	=
2	6	1	7
1	2	-1	-1
5	7	x	9

Y el sistema informara al usuario que corrija los datos:



El proceso de resolución consiste en hacer que el método numérico coloque 0 en la parte inferior de la diagonal principal de la matriz aumentada.

Antes de convertir en ceros los valores debajo de la diagonal de la matriz aumentada:

+2X	+6Y	+1Z	= +7
+1X	+2Y	-1Z	= -1
+5X	+7Y	-4Z	= +9

2	6	1	7
1	2	-1	-1
5	7	-4	9

Fragmento de código:

El sistema realiza la conversión de cada columna debajo de la diagonal de la matriz aumentada, determinando que valor debe de utilizar para restar con el pivote que es la primera fila y así convertir en 0 la primera columna posteriormente se imprime el log de sucesos de la operación.

```
Write_log("Resolviendo sistema de ecuaciones por metodo de eliminación Gaussiana")

Dim vSigno As Double = _x1 * _x2
Write_log("")
Write_log("Cero en la columna 1 fila 2")
Write_log("F2 = F2 " + IIf(vSigno >= 0, "-", "+") + Replace(_x2.ToString, "-", "") + "/" + Replace(_x1.ToString, "-", "") + " F1")
Write_log("")

Dim factor As Double = Math.Abs(_x2 / _x1)
If (_x1 = 0) Then
    Write_log("Error denominador es Cero")
    MsgBox("Error denominador es Cero")
    Exit Try
End If

_x2 = FormatNumber(_x2 + IIf(vSigno >= 0, -factor * _x1, +factor * _x1), 4)
_y2 = FormatNumber(_y2 + IIf(vSigno >= 0, -factor * _y1, +factor * _y1), 4)
_z2 = FormatNumber(_z2 + IIf(vSigno >= 0, -factor * _z1, +factor * _z1), 4)
_n2 = FormatNumber(_n2 + IIf(vSigno >= 0, -factor * _n1, +factor * _n1), 4)

PL_Imprimir_Matriz(_x1, _x2, _x3, _y1, _y2, _y3, _z1, _z2, _z3, _n1, _n2, _n3, "|")
```

La matriz quedaría de la siguiente forma:

2	6	1	7
0	-1	-1.5	-4.5
5	7	-4	9

Y en un segundo paso finaliza de colocar el siguiente cero en la columna numero 1

```
vSigno = _x1 * _x3

Write_log("Cero en la columna 1 fila 3")
Write_log("F3 = F3 " + IIf(vSigno >= 0, "-", "+") + Replace(_x3.ToString, "-", "") + "/" + Replace(_x1.ToString, "-", "") + " F1")
Write_log("")

factor = Math.Abs(_x3 / _x1)
If (_x1 = 0) Then
    Write_log("Error denominador es Cero")
    MsgBox("Error denominador es Cero")
    Exit Try
End If

_x3 = FormatNumber(_x3 + IIf(vSigno >= 0, -factor * _x1, +factor * _x1), 4)
_y3 = FormatNumber(_y3 + IIf(vSigno >= 0, -factor * _y1, +factor * _y1), 4)
_z3 = FormatNumber(_z3 + IIf(vSigno >= 0, -factor * _z1, +factor * _z1), 4)
_n3 = FormatNumber(_n3 + IIf(vSigno >= 0, -factor * _n1, +factor * _n1), 4)

PL_Imprimir_Matriz(_x1, _x2, _x3, _y1, _y2, _y3, _z1, _z2, _z3, _n1, _n2, _n3, "|")
```

Así quedaría la matriz luego del proceso:

2	6	1	7
0	-1	-1.5	-4.5
0	-8	-6.5	-8.5

Las operaciones que utilizó el sistema para resolver son las siguientes:

Cero en la columna 1 fila 2
 $F2 = F2 - 1/2 F1$

Cero en la columna 1 fila 3
 $F3 = F3 - 5/2 F1$

El siguiente paso en el proceso es hacer 0 el valor de la columna y3 para que la matriz tenga ceros bajo su diagonal principal

El código utilizado para este proceso es el siguiente:

```
vSigno = _y2 * _y3
Write_log("Cero en la columna 2 fila 3")
Write_log("F3 = F3 " + IIf(vSigno >= 0, "-", "+") + Replace(_y3.ToString, "-", "") + "/" + Replace(_y2.ToString, "-", "") + " F2")
Write_log("")

factor = Math.Abs(_y3 / _y2)
If (_y2 = 0) Then
    Write_log("Error denominador es Cero")
    MsgBox("Error denominador es Cero")
    Exit Try
End If

_x3 = FormatNumber(_x3 + IIf(vSigno >= 0, -factor * _x2, +factor * _x2), 4)
_y3 = FormatNumber(_y3 + IIf(vSigno >= 0, -factor * _y2, +factor * _y2), 4)
_z3 = FormatNumber(_z3 + IIf(vSigno >= 0, -factor * _z2, +factor * _z2), 4)
_n3 = FormatNumber(_n3 + IIf(vSigno >= 0, -factor * _n2, +factor * _n2), 4)

Write_log("Mostrando matriz escalonada:")
PL_Imprimir_Matriz(_x1, _x2, _x3, _y1, _y2, _y3, _z1, _z2, _z3, _n1, _n2, _n3, "|")
```

El cual utilizara la siguiente operación:

Cero en la columna 2 fila 3
 $F3 = F3 - 8/1 F2$

Dará como resultado la matriz siguiente

	2		6		1		7	
	0		-1		-1.5		-4.5	
	0		0		5.5		27.5	

Al finalizar el proceso se revisa como quedaron los valores correspondientes a la ultima fila de la matriz para determinar qué tipo de solución tiene el sistema:

```
'Valido las soluciones
If _z3 = 0 And _n3 = 0 Then
    Write_log("El sistema de ecuaciones tiene infinitas soluciones...")
    MsgBox("El sistema de ecuaciones tiene infinitas soluciones")
    Write_log("")
    Exit Try
End If

If _z3 = 0 And _n3 <> 0 Then
    Write_log("El sistema de ecuaciones no tiene solucion...")
    MsgBox("El sistema de ecuaciones no tiene solucion")
    Write_log("")
    Exit Try
End If
```

El sistema tendrá soluciones infinitas si la ultima fila contiene solo ceros.

El sistema tendrá ninguna solución si solo existe cualquier valor distinto de cero en la columna numero 4 posición 4.

De lo contrario el sistema determina que si hay soluciones para el problema propuesto y realiza la extracción de los resultados:

```
'Presento las soluciones de la ecuacion
Write_log("Presentando las soluciones:.....")
Write_log("")

txt_RZ.Text = FormatNumber(_n3 / _z3, 4).ToString
Write_log("Determino el valor de Z: " + txt_RZ.Text)
If (_z3 = 0) Then
    Write_log("Error denominador es Cero")
    MsgBox("Error denominador es Cero")
    Exit Try
End If
txt_RV.Text = FormatNumber((( _n2 - _z2 * (_n3 / _z3)) / _y2), 4).ToString
Write_log("Determino el valor de Y: " + txt_RV.Text)
If (_y2 = 0) Then
    Write_log("Error denominador es Cero")
    MsgBox("Error denominador es Cero")
    Exit Try
End If
txt_RX.Text = FormatNumber((( _n1 - _y1 * (( _n2 - _z2 * (_n3 / _z3)) / _y2) - _z1 * (_n3 / _z3)) / _x1), 4).ToString
Write_log("Determino el valor de X: " + txt_RX.Text)
If (_x1 = 0) Then
    Write_log("Error denominador es Cero")
    MsgBox("Error denominador es Cero")
    Exit Try
End If
```

Presentación de los resultados:

El algoritmo puede presentar de dos formas los resultados de la ejecución del programa los cuales son los siguientes:

Presentación de los valores de las soluciones:

Solución:		
X	=	10.0000
Y	=	-3.0000
Z	=	5.0000

El cual nos dará las soluciones utilizando 4 posiciones decimales.

El segundo método es con el log de sucesos en el que podemos ver a detalle cada uno de los pasos que el programa utilizó para llegar a la solución y el cual podremos copiar y pegar en algún procesador de texto para analizarlo o simplemente visualizarlo en el espacio destinado:

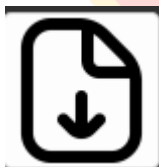
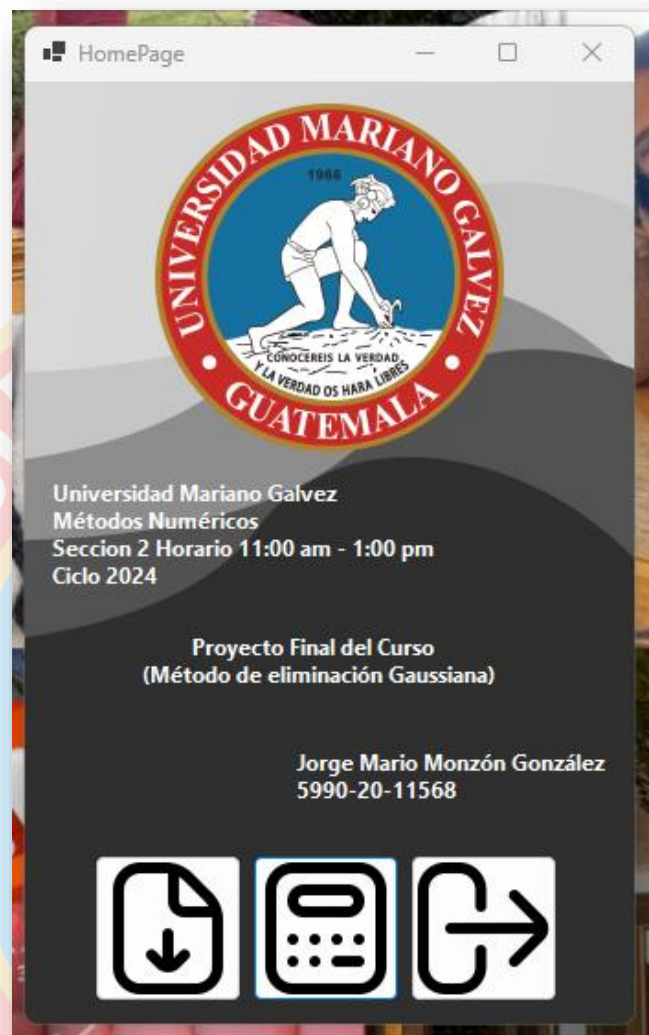
Pasos:

Iniciando la ejecución.....

Seteando Ceros en casillas vacías de X
Seteando Ceros en casillas vacías de Y
Seteando Ceros en casillas vacías de Z
Seteando Ceros en casillas vacías de las igualdades
Seteando Ceros con Éxito....

Validar si los valores ingresados son

Partes de la interfaz del programa:



Descargar instructivo



Utilizar calculadora



Salir

Ecuaciones Lineales

Método de Eliminación Gaussiana

Ingreso de Datos:

X	Y	Z	=
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>




Pasos:

Solución:

X =

Y =

Z =



Limpiar Calculadora



Realizar Calculo



Salir a menú principal

Problemas Resueltos:

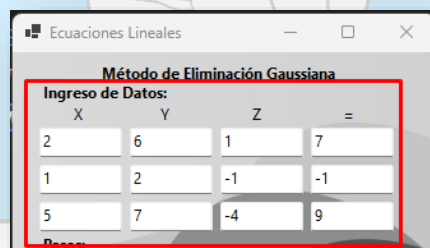
Para comprobar la funcionalidad del programa se presentan 3 problemas en los cuales se podrá observar la capacidad de calculo y el error que pueda tener el mismo.

Primer Escenario (Sistema con única solución)

Se presenta el siguiente problema:

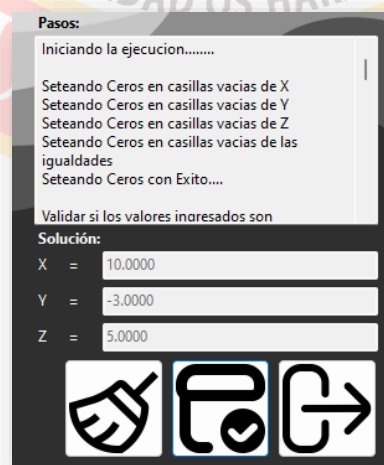
$$\begin{array}{rrcr} +2X & +6Y & +1Z & = +7 \\ +1X & +2Y & -1Z & = -1 \\ +5X & +7Y & -4Z & = +9 \end{array}$$

Se ingresan los valores en la calculadora:



X	Y	Z	=
2	6	1	7
1	2	-1	-1
5	7	-4	9

Al presionar el botón calcular se observa la generación de log y de los resultados:



Pasos:

- Iniciando la ejecución.....
- Seteando Ceros en casillas vacías de X
- Seteando Ceros en casillas vacías de Y
- Seteando Ceros en casillas vacías de Z
- Seteando Ceros en casillas vacías de las igualdades
- Seteando Ceros con Éxito....

Validar si los valores ingresados son

Solución:

X = 10.0000

Y = -3.0000

Z = 5.0000

Below the solution, there are three icons: a hand pointing to a button, a document with a checkmark, and a circular arrow.

Segundo Escenario (Sistema con Infinitas Soluciones)

Se presenta el siguiente problema:

$$\begin{array}{rrcr} +1X & +2Y & +3Z & = +4 \\ +5X & +6Y & +7Z & = +8 \\ +9X & +10Y & +11Z & = +12 \end{array}$$

Se ingresan los valores en la calculadora:

Ingreso de Datos:			
X	Y	Z	=
1	2	3	4
5	6	7	8
9	10	11	12




Al presionar el botón calcular se observa la generación de log y de los resultados:

Pasos:
| 0 | 0 | 0 | 0 |

El sistema de ecuaciones tiene infinitas soluciones...

Fin de la ejecución.....

Solución:
X =
Y =
Z =

Microsoft.VisualBasic.Core

El sistema de ecuaciones tiene infinitas soluciones

Aceptar

Tercer Escenario (Sistema sin soluciones)

Se presenta el siguiente problema:

$$\begin{array}{rrcr} +2X & -6Y & +7Z & = +15 \\ +1X & -2Y & +5Z & = +10 \\ +0X & +2Y & +3Z & = +4 \end{array}$$

Se ingresan los valores en la calculadora:

Ingreso de Datos:			
X	Y	Z	=
2	-6	7	15
1	-2	5	10
0	2	3	4

Al presionar el botón calcular se observa la generación de log y de los resultados:

Pasos:

Iniciando la ejecución.....

Seteando Ceros en casillas vacias de X
Seteando Ceros en casillas vacias de Y
Seteando Ceros en casillas vacias de Z
Seteando Ceros en casillas vacias de las igualdades
Seteando Ceros con Exito....

Validar si los valores ingresados son

Solución:

X =

Y =

Z =

Microsoft.VisualBasic.Core

El sistema de ecuaciones no tiene solución

Aceptar

Análisis y discusión

Se puede observar que para cada uno de los escenarios el sistema entregó resultados satisfactorios y eficientes ya que el algoritmo posee una precisión de 4 cifras decimales lo cual permite que los resultados tengan una buena precisión, además de poder determinar de forma ágil si el sistema de ecuaciones propuesto en cada escenario tiene solución o si no la tiene y así mismo si el sistema cuenta con soluciones infinitas, al seguir al puntualmente el algoritmo que propone el método de solución por eliminación gaussiana el sistema puede en pocos pasos determinar estos resultados y ser eficiente ya que no tiene alta latencia en la obtención de los resultados y permite al usuario observar cada uno de los pasos que ejecuto el programa, adicional gracias a su precisión de 4 decimales podemos utilizar números reales y así utilizar su precisión,

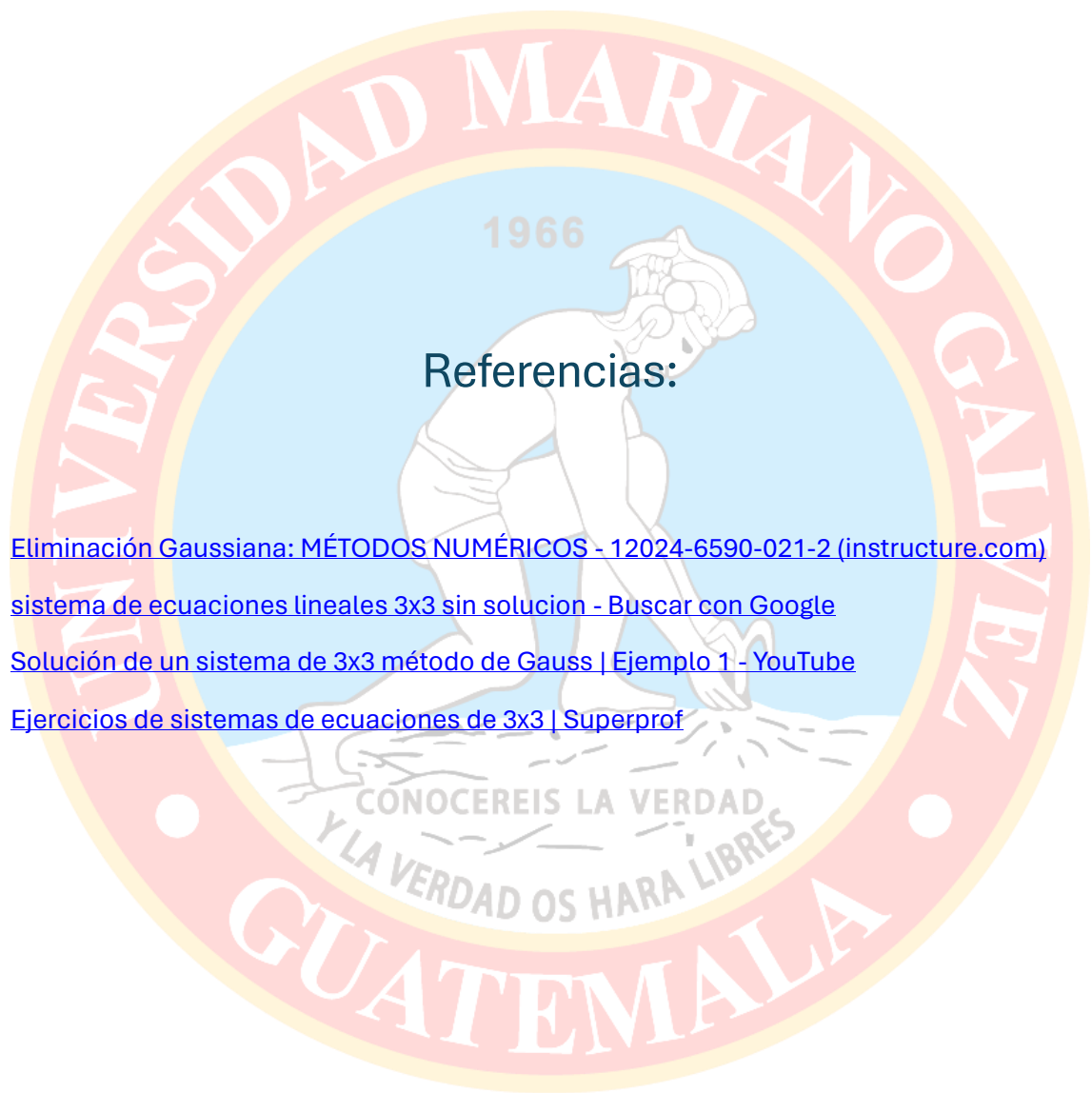
Una de las primeras dificultades fue la captura de el log de sucesos ya que en general es la parte que más tiempo lleva estar formateando cada uno de los mensajes.

Otra dificultad fue colocar validaciones de entrada ya que el programa debe tener alta calidad y no debe permitir el ingreso a escenarios no controlados por lo que se agregaron validaciones para los campos de ingreso de datos.

La dificultad final fue determinar la precisión del programa ya que si se dejaba libre el campo el mismo iba a escribir una cantidad grande de dígitos luego del punto decimal en los logs y esto ocasionaría que el programa fuera incomprensible por lo que limite a 4 cifras significativas el programa y así las respuestas son comprensibles.

Conclusión:

- 1) Los métodos numéricos nos permiten aproximar soluciones a problemas matemáticas con buena precisión y con el uso de las computadoras podemos ampliar esa precisión.
- 2) La implementación de programas informáticos para la solución de métodos numéricos provee herramientas que simplifican las tareas de los ingenieros ya que no tienen que hacer cálculos manuales.
- 3) Al utilizar estos programas con capacidad de resolver este tipo de problemas que pueden volverse complejos le trasladamos a la computadora el trabajo y eso nos hace más eficientes ya que podemos ahorrar tiempos en la solución de estos problemas.



Referencias:

[Eliminación Gaussiana: MÉTODOS NUMÉRICOS - 12024-6590-021-2 \(instructure.com\)](#)

[sistema de ecuaciones lineales 3x3 sin solución - Buscar con Google](#)

[Solución de un sistema de 3x3 método de Gauss | Ejemplo 1 - YouTube](#)

[Ejercicios de sistemas de ecuaciones de 3x3 | Superprof](#)