



EDUCACIÓN

SECRETARÍA DE EDUCACIÓN PÚBLICA



INSTITUTO TECNOLÓGICO DE COMITAN

Alumnos:

- Ruedas Velasco Pedro Eduardo__19700073.
- Hernández Méndez Levi Magdiel__19700039.
- Molina Cifuentes Adriel David__19700061.
- Panti Ordoñez Sergio Ismael__19700065.

Docente: Vera Guillen José Flavio.

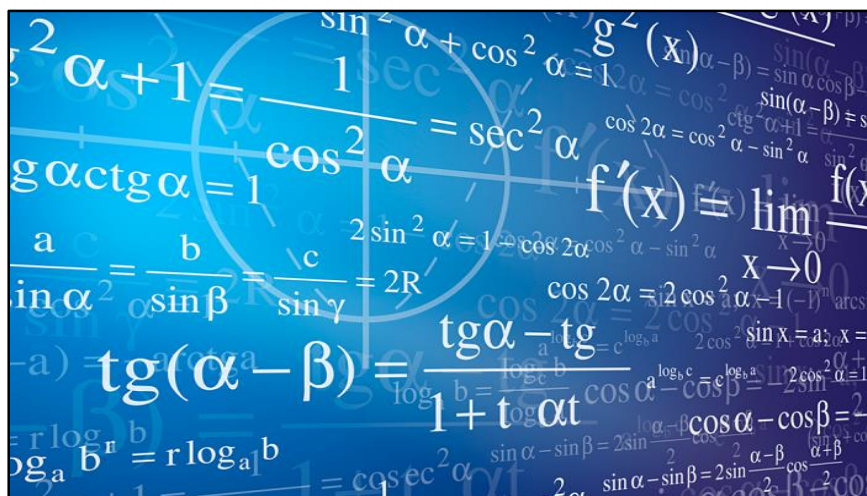
Materia: Métodos Numéricos.

Semestre: Cuarto

Grupo: "A"

Actividad: PA6_ Códigos de los métodos de eliminación en Java.

Comitán de Domínguez Chiapas, a 02 de Mayo del 2021.



Método de eliminación en Java "Gauss Jordan".

Código Empleado:

```
13  */
14  public class Gauss_Jordan extends javax.swing.JFrame {
15      //Numero de incognitas en la ecuacion
16      int n;
17      DefaultTableModel modelo =new DefaultTableModel();
18      Boolean pd=false;
19
20      public int getN() {
21          return n;
22      }
23      public void setN(int n) {
24          this.n = n;
25      }
26      /**
27       * Creates new form Gauss_Jordan
28       */
29      public Gauss_Jordan() {
30          initComponents();
31          this.setLocationRelativeTo(null);
32      }
```

```
33      public double[] resolver(double m[], double r[]) {
34          /* recuerde que el metodo de Gauss Jordan trabaja con la idea de convertir
35             la matriz aumentada en la matriz identidad*/
36          int n = this.getN();
37          for (int i = 0; i < n; i++) {
38              double d, c = 0;
39              d = m[i][i]; // seleccionamos el pivote
40              txta.append(Double.toString(1 / d) + "**fila" + (i + 1) + "\n");
41              // muestra en el area de texto el pivote seleccionado
42              for (int j = 0; j < n; j++) { // pasamos a convertir en 1 al pivote seleionado
43                  m[i][j] = ((m[i][j]) / d);
44                  r[i] = ((r[i]) / d);
45                  // paso a mostrar las operaciones realizadas en la matriz aumentada
46                  for (int j = 0; j < n; j++) {
47                      for (int k = 0; k < n; k++) {
48                          txta.append(Double.toString(m[j][k]) + "\t");
49                      }
50                      txta.append("\t" + Double.toString(r[j]) + "\n");
51                  }
52                  txta.append("\n\n"); // fin paso a motrar las operaciones realizadas en la matriz a
53              }
```

```
54         for (int x = 0; x < n; x++) {
55             if (i != x) {
56                 c = m[x][i];
57                 txta.append("-" + Double.toString(c) + " * fila" + (i + 1) +
58                     "+ fila" + (x + 1) + "\n");
59                 // muestra en pantalla las operaciones que se realizaran por fila
60                 for (int y = 0; y < n; y++) { // se hace cero a todos los elemntos de la
61                     //columna que no sean el pivote
62                         m[x][y] = m[x][y] - c * m[i][y];
63                     }
64                 r[x] = r[x] - c * r[i];
65                 // paso a mostrar las opraciones realizadas en la matriz aumentada
66                 for (int j = 0; j < n; j++) {
67
68                     for (int k = 0; k < n; k++) {
69                         txta.append(Double.toString(m[j][k]) + "\t");
70                     }
71                     txta.append("| \t" + Double.toString(r[j]) + "\n");
72                 }
73                 txta.append("\n\n"); // fin paso a motrar las opraciones realizadas
74                 //en la matriz aumentada.
75             }
76         }
77     }
78     return r; // retorna la solucion del sistema
79 }
```

```
233 private void salirActionPerformed(java.awt.event.ActionEvent evt) {
234     // TODO add your handling code here:
235     dispose();
236     JOptionPane.showMessageDialog(null, "Vuelva Pronto");
237 }
238
239 private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
240     // TODO add your handling code here:
241     try {
242         // Se convierte el dato de string a entero
243         this.setN(n=Integer.parseInt(txt.getText()));
244         n=this.getN();
245         if (n<=1) {
246             // Si el numero de incognitas es menor o igual a 1
247             //se generara un error de la clase exception
248             throw new Exception("Ha ingresado un numero de incognitas invalido");
249         }
250         //Si el numero de incognitas es mayor o igual a 2 se generara una matriz
251         Object columna[]=new Object[n+1];
```

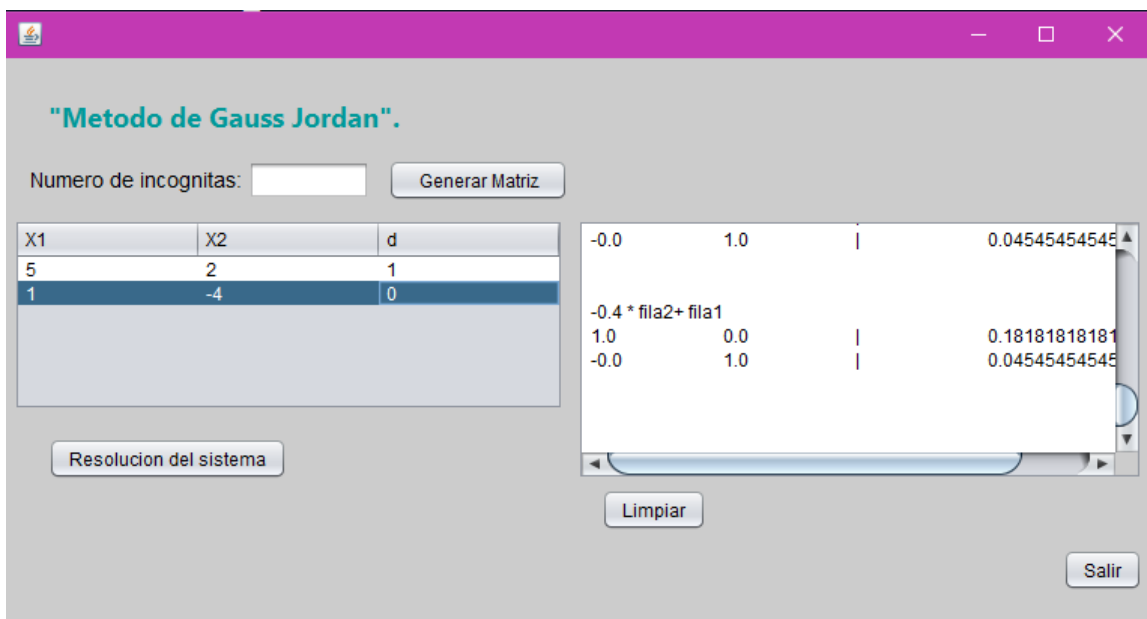
```
252         for (int i = 0; i < n+1; i++) {
253             if (i<n) {
254                 columna[i]="X"+(i+1);
255             }else{
256                 columna[i]="d";
257             }
258         }
259         modelo =new DefaultTableModel(columna,n);
260         tabla.setModel(modelo);
261         txt.setText("");
262     } catch (Exception e) {
263         JOptionPane.showMessageDialog(null, e.getMessage());
264     }
265 }
```

```

266
267 private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
268     // TODO add your handling code here:
269     txta.setText(" ");
270 }
271
272 private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {
273     // TODO add your handling code here:
274     try {
275         int n=this.getN();
276         double m[][] =new double[n][n];
277         //Almacena los coeficientes para la matriz aumentada
278         double r[] = new double [n];
279         //Almacena el valor de la solucion de cada ecuacion
280
281         for (int i = 0; i < n; i++) {
282             for (int j = 0; j < n; j++) {
283                 m[i][j]=Double.parseDouble(String.valueOf(tabla.getValueAt(i, j)));
284             }
285
286             r[i]=Double.parseDouble(String.valueOf(tabla.getValueAt(i, n)));
287         }
288         r=this.resolver(m,r);
289     } catch (Exception e) {
290         JOptionPane.showMessageDialog(null, "Error en la lectura de datos");
291     }
292 }

```

Resultado en el programa:



"Metodo de Gauss Jordan".

Numero de incognitas: Generar Matriz

X1	X2	d
5	2	1
1	-4	0

Resolucion del sistema

Limpiar

Salir

Visualización de la matriz aumentada:

```

-0.0      1.0      |      0.04545454545
-0.4 * fila2+ fila1
1.0      0.0      |      0.18181818181
-0.0      1.0      |      0.04545454545

```

Resultado a mano:

Método de Gauss Jordan.

$$5x_1 + 2x_2 = 1$$

$$x_1 - 4x_2 = 0$$

Solución : se escribe la matriz y se transforma

$$\left[\begin{array}{cc|c} 5 & 2 & 1 \\ 1 & -4 & 0 \end{array} \right] \xrightarrow{M_1(\frac{1}{5})} \left[\begin{array}{cc|c} 1 & \frac{2}{5} & \frac{1}{5} \\ 1 & -4 & 0 \end{array} \right] \xrightarrow{M_{2,1}(-1)} \left[\begin{array}{cc|c} 1 & \frac{2}{5} & \frac{1}{5} \\ 0 & -\frac{12}{5} & -\frac{1}{5} \end{array} \right] \xrightarrow{M_2(-\frac{5}{12})}$$

$$\left[\begin{array}{cc|c} 1 & \frac{2}{5} & \frac{1}{5} \\ 0 & 1 & \frac{1}{22} \end{array} \right] \xrightarrow{M_{1,2}(-\frac{2}{5})} \left[\begin{array}{cc|c} 1 & 0 & \frac{2}{11} \\ 0 & 1 & \frac{1}{22} \end{array} \right]$$

la solución única es:

$$x_1 = \frac{2}{11}$$

$$x_2 = \frac{1}{22}$$

comprobación

$$5\left(\frac{2}{11}\right) + 2\left(\frac{1}{22}\right) = 1 \quad \rightarrow 1 = 1$$

$$\left(\frac{2}{11}\right) - 4\left(\frac{1}{22}\right) = 0 \quad \rightarrow 0 = 0$$

Método de eliminación en Java "Gauss Seidel".

Código Realizado:

```
Source Design History
6 package gauss_seidel;
7
8 import javax.swing.JOptionPane;
9 import javax.swing.table.DefaultTableModel;
10
11 /**
12  *
13  * @author Levi -KC-
14  */
15 public final class Main extends javax.swing.JFrame {
16
17
18
19 public Main() {
20     initComponents();
21     this.setLocationRelativeTo( null );
22 }
23
24 public void refrescar() {
25     txtCant.setText("");
26     txtEpsilon.setText("");
27     lblIteraciones.setText("");
28     lblTitulo.setText("");
29     DefaultTableModel model = (DefaultTableModel) tblIteraciones.getModel();
30     int f = tblIteraciones.getRowCount();
31     for (int i = f-1; i >= 0; i--) {
32         model.removeRow(i);
33     }
34     DefaultTableModel modelo = (DefaultTableModel) tblResultado.getModel();
35     int fi = tblResultado.getRowCount();
36     for (int i = fi-1; i >= 0; i--) {
37         modelo.removeRow(i);
38     }
39 }
40
41 private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
42
43     Double e = Double.parseDouble(txtEpsilon.getText());
44     Integer n = Integer.parseInt(txtCant.getText());
45     Integer a = n+1;
46     Double[][] M = new Double[10][10];
47     Double[][] Iter = new Double[999][999];
48     Double [] V = new Double[100];
49
50     for (int f = 1; f <= n; f++) {
51         for (int c = 1; c <= a; c++) {
52             if (c == a) {
53                 M[f][c] = Double.parseDouble(JOptionPane.showInputDialog("Introduce el termino independiente de la ecuacion"));
54                 /*JOptionPane.showMessageDialog(null,"variable de la matriz "+f+" "+c+" : "+ M[f][c]);*/
55             }else {
56                 M[f][c] = Double.parseDouble(JOptionPane.showInputDialog("Introduce los coeficientes de la "+ f +" ecuacion (un"));
57                 /*JOptionPane.showMessageDialog(null,"variable de la matriz "+f+" "+c+" : "+ M[f][c]);*/
58             }
59         }
60     }
61 }
62 }
```

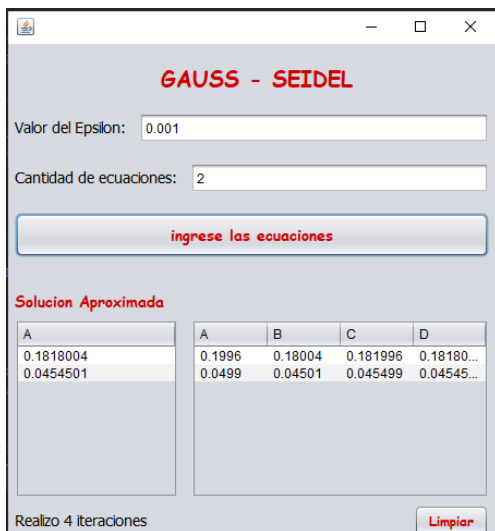


```
203
204 Double may;
205 Integer ban=0;
206 for (int f = 1; f <= n; f++) {
207     may = M[f][f];
208     for (int c = 1; c < n+1; c++) {
209         if ((Math.abs(may)) < (Math.abs(M[f][c]))) {
210             ban = 1;
211         }
212     }
213 }
214
215 if (ban == 1) {
216     JOptionPane.showMessageDialog(null, "El sistema de ecuaciones no converge. Intente nuevamente");
217     this.refrescar();
218 } else {
219     JOptionPane.showMessageDialog(null, "El sistema de ecuaciones converge");
220
221     for (int f = 1; f <= n; f++) {
222         V[f] = Double.parseDouble(JOptionPane.showInputDialog("Introduce los valores iniciales"));
223     }
```

```
225 Integer inter = 0, c = 1;
226 Double mayor, suma, res, er;
227 do {
228     mayor = 0.0;
229     for (int i = 1; i <= n; i++) {
230         suma = 0.0;
231         for (int j = 1; j <= n; j++) {
232             if (j != i) {
233                 suma += M[i][j] * V[j];
234             }
235         }
236         res = (M[i][a] - suma) / M[i][i];
237         er = Math.abs(V[i] - res);
238         if (er > mayor) {
239             mayor = er;
240         }
241         V[i] = res;
242         Iter[i][c] = res;
243     }
244     c++;
245     inter++;
246 } while (mayor >= e);
```

```
247
248 lblTitulo.setText("Solucion Aproximada");
249 lblIteraciones.setText("Realizo "+inter+" iteraciones");
250
251 DefaultTableModel modelo = (DefaultTableModel) tblResultado.getModel();
252 modelo.setRowCount(n);
253 modelo.setColumnCount(1);
254 for (int i = 0; i < n; i++) {
255     tblResultado.setValueAt(V[i+1], i, 0);
256 }
257 DefaultTableModel model = (DefaultTableModel) tblIteraciones.getModel();
258 model.setRowCount(n);
259 model.setColumnCount(inter);
260 for (int i = 0; i < n; i++) {
261     for (int j = 0; j < inter; j++) {
262         tblIteraciones.setValueAt(Iter[i+1][j+1], i, j);
263     }
264 }
265
266
267
268
269 }
```


Resultado en Java:



GAUSS - SEIDEL

Valor del Epsilon: 0.001

Cantidad de ecuaciones: 2

ingrese las ecuaciones

Solucion Aproximada

A	B	C	D
0.1818004	0.1996	0.181996	0.18180...
0.0454501	0.0499	0.045499	0.04545...

Realizo 4 iteraciones

Limpiar

Resultado a mano:

Metodo Gauss - Seidel

$$5x + 2y = 1$$

$$x - 4y = 0$$

$$\epsilon_{adm} = 0.001$$

1. $|5| > |2|$
 $|1-4| > |1|$

2. $x = (1 - 2y) / 5$
 $y = x / 4$

3. $x_0 = 1, y_0 = 2$

4. $x_1 = (1 - 2y_0) / 5 = (1 - 2 \cdot 2) / 5 = -3/5$
 $y_1 = x_1 / 4 = (-3/5) / 4 = -3/20$

5. $x_2 = (1 - 2y_1) / 5 = (1 - 2(-3/20)) / 5 = 13/50$
 $y_2 = x_2 / 4 = (13/50) / 4 = 13/200$

6. $x_3 = (1 - 2y_2) / 5 = (1 - 2(13/200)) / 5 = 87/500$
 $y_3 = x_3 / 4 = (87/500) / 4 = 87/2000$

7. $x_4 = (1 - 2y_3) / 5 = (1 - 2(87/2000)) / 5 = 913/5000$
 $y_4 = x_4 / 4 = (913/5000) / 4 = 913/20000$

$x_4 = 0.1826$
 $y_4 = 0.04565$

$x = 2/11 = 0.1818$
 $y = 1/22 = 0.04545$



"Conclusión"

En los resultados obtenidos de la ecuación introducida en el programa en java y el elaborado a mano, con el mismo problema introducido en ambos.

Se llegó a una conclusión de que, ambos resultados coinciden, pero en el programa en java, al realizar el ejercicio se hace en menos tiempo y sin cometer errores, mientras que elaborado a mano lleva más tiempo y con más probabilidades de cometer errores.