TUGAS BESAR 1 IF2123
Aljabar Linear dan Geometri
2022/2023

KELOMPOK 41 - Go Yoon Jung

Brian Kheng	**13521049**
Farizki Kurniawan	**13521082**
Frankie Huang	**13521092**

SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA
INSTITUT TEKNOLOGI BANDUNG

BAB 1
DESKRIPSI MASALAH

Sistem persamaan linier (SPL) banyak ditemukan di dalam bidang sains dan rekayasa. Sembarang SPL dapat diselesaikan dengan beberapa metode, yaitu metode eliminasi Gauss, metode eliminasi Gauss-jordan, metode matriks balikan ($x = A^{-1}b$), dan kaidah *Cramer* (khusus untuk SPL dengan *n* peubah dan *n* persamaan). Solusi sebuah SPL mungkin tidak ada, banyak (tidak berhingga), atau hanya satu (unik/tunggal).

Di dalam Tugas Besar 1 ini, kami membuat *library* aljabar linier dalam Bahasa Java. Library tersebut berisi fungsi-fungsi seperti eliminasi Gauss, eliminasi Gauss-Jordan, menentukan balikan matriks, menghitung determinan, kaidah *Cramer*. Selanjutnya *library* tersebut digunakan di dalam program Java untuk menyelesaikan berbagai persoalan yang dimodelkan dalam bentuk SPL, menyelesaikan persoalan interpolasi, dan persoalan regresi.

BAB 2
TEORI SINGKAT

2.1. Sistem Persamaan Linier

2.1.1 Metode Eliminasi Gauss

Metode yang dinamai oleh matematikawan Carl Friedrich Gauss (1777-1855) ini merupakan salah satu algoritma dalam penyelesaian SPL. Metode eliminasi Gauss dalam penyelesaian SPL membentuk matriks augmented menjadi matriks eselon baris dengan menggunakan operasi baris elementer. Yang nantinya akan dilakukan teknik penyulihan mundur untuk mendapatkan nilai x.

$$\begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} & b_1 \\ a_{21} & a_{22} & \dots & a_{2n} & b_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} & b_n \end{bmatrix} \sim\text{OBE}\sim \begin{bmatrix} 1 & * & * & \dots & * & * \\ 0 & 1 & * & \dots & * & * \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \vdots & 1 & * \end{bmatrix}$$

2.1.2. Metode Eliminasi Gauss Jordan

Metode ini merupakan variasi dari eliminasi Gauss yang dijelaskan oleh Wilhelm Jordan pada tahun 1888. Metode eliminasi Gauss Jordan membentuk matriks augmented menjadi matriks eselon baris tereduksi (elemen di atas dan di bawah 1 utama bernilai 0). Sehingga, nantinya tidak perlu dilakukan teknik penyulihan mundur untuk mendapatkan nilai x.

$$\begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} & b_1 \\ a_{21} & a_{22} & \dots & a_{2n} & b_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} & b_m \end{bmatrix} \sim\text{OBE}\sim \begin{bmatrix} 1 & 0 & 0 & \dots & 0 & * \\ 0 & 1 & 0 & \dots & 0 & * \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \vdots & 1 & * \end{bmatrix}$$

2.1.3. Metode Matriks Balikan

Metode penentuan SPL dengan menggunakan matriks balikan hanya dapat digunakan pada matriks persegi dan juga ketika determinan ≠ 0. Pada metode ini, solusi SPL didapatkan dengan $x = A^{-1} b$.

2.1.4. Kaidah Cramer

Menurut Kaidah Cramer, jika Ax = b adalah SPL yang terdiri dari n persamaan linier dengan n peubah (variable) sedemikian sehingga det(A) ≠ 0, maka SPL tersebut memiliki solusi yang unik, yaitu

$$x_1 = \frac{\det(A_1)}{\det(A)}, \quad x_2 = \frac{\det(A_2)}{\det(A)}, \quad \ldots, \quad x_n = \frac{\det(A_n)}{\det(A)}$$

dimana A_i adalah matriks yang diperoleh dengan mengganti entri pada kolom ke-i dari A dengan entri dari matriks b.

2.2. Determinan

Determinan adalah sebuah abstraksi yang melambangkan suatu nilai yang bisa didapatkan dari sebuah matriks persegi. Determinan dari suatu matriks persegi A umumnya dilambangkan dengan det(A). Terdapat beberapa cara untuk menentukan determinan dari suatu matriks persegi, dua diantaranya adalah dengan metode reduksi baris dan dengan metode ekspansi kofaktor.

2.2.1. Metode Reduksi Baris

Determinan matriks A dapat diperoleh dengan melakukan OBE pada matriks A sampai diperoleh matriks segitiga (segitiga bawah atau atas).

$$[A] \overset{\text{OBE}}{\sim} [\text{matriks segitiga bawah}]$$

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix} \overset{\text{OBE}}{\sim} \begin{bmatrix} a'_{11} & a'_{12} & \cdots & a'_{1n} \\ 0 & a'_{22} & \cdots & a'_{2n} \\ \vdots & \vdots & \cdots & a'_{3n} \\ 0 & 0 & 0 & a'_{nn} \end{bmatrix}$$

Sehingga didapatkan

$$\det(A) = (-1)^p \, a'_{11} a'_{22} \ldots a'_{nn}$$

dimana p menyatakan banyaknya operasi pertukaran baris di dalam OBE.

2.2.2. Metode Ekspansi Kofaktor

Didefinisikan sebuah matriks persegi A sebagai berikut

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix}$$

Definisikan notasi M_{ij} sebagai minor dari entri a_{ij}, yaitu determinan dari sub-matriks yang elemen-elemennya adalah elemen matriks A yang tidak berada pada baris i dan kolom j.

Lalu didefinisikan juga C_{ij} sebagai kofaktor dari entri a_{ij}, yaitu

$$C_{ij} = (-1)^{i+j} M_{ij}$$

Maka, determinan dari matriks A dapat ditentukan dengan salah satu dari persamaan berikut.

$$\det(A) = a_{11}C_{11} + a_{12}C_{12} + \ldots + a_{1n}C_{1n}$$
$$\det(A) = a_{21}C_{21} + a_{22}C_{22} + \ldots + a_{2n}C_{2n}$$
$$\vdots$$
$$\det(A) = a_{n1}C_{n1} + a_{n2}C_{n2} + \ldots + a_{nn}C_{nn}$$

Secara baris

$$\det(A) = a_{11}C_{11} + a_{21}C_{21} + \ldots + a_{n1}C_{n1}$$
$$\det(A) = a_{12}C_{12} + a_{22}C_{22} + \ldots + a_{n2}C_{n2}$$
$$\vdots$$
$$\det(A) = a_{1n}C_{1n} + a_{2n}C_{2n} + \ldots + a_{nn}C_{nn}$$

Secara kolom

2.3. Balikan Matriks

Invers dari suatu matriks adalah suatu matriks yang jika dikalikan dengan matriks semula akan menghasilkan matriks identitas. Agar suatu matriks dapat memiliki determinan, harus ada 2 syarat yang harus dipenuhi, yaitu determinan matriks tersebut tidak boleh nol dan harus berbentuk persegi. Terdapat beberapa cara untuk mencari balikan matriks, dua diantaranya adalah dengan menggunakan matriks adjoin dan eliminasi gauss jordan.

2.3.1. Metode Matriks Adjoin

Didefinisikan suatu matriks sebagai berikut

$$A = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{bmatrix}$$

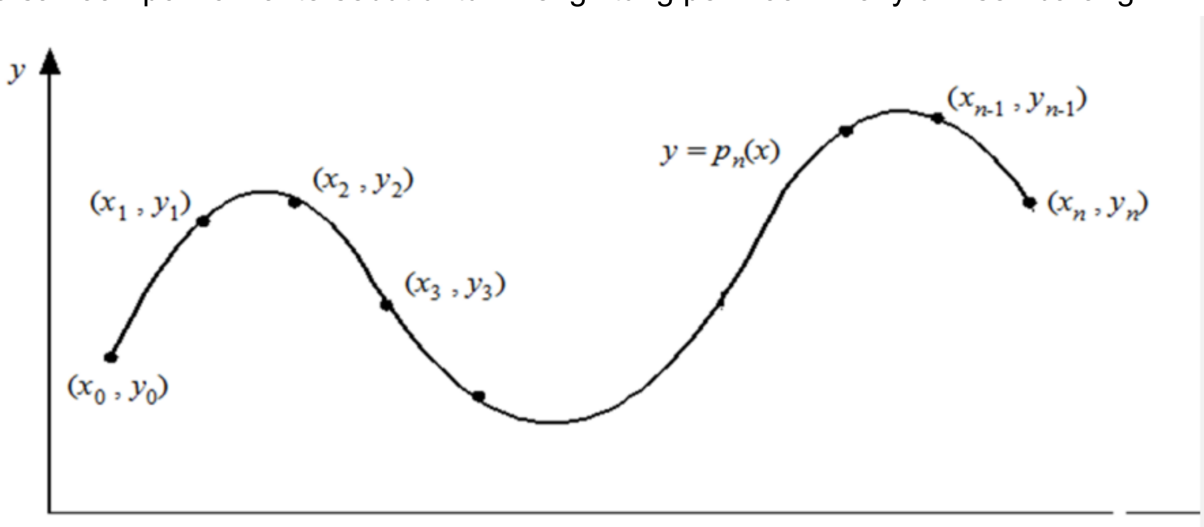
Maka entri kofaktor pada baris *i* dan kolom *j* adalah -1 pangkat i+j dikali dengan determinan dari matriks minor pada baris *i* dan kolom *j*, yaitu matriks semula yang tidak memiliki elemen pada baris *i* dan kolom *j*. Setelah kofaktor dari suatu matriks didapatkan, balikan matriks dapat dicari dengan mentranspose matriks adjoin dan mengalikan tiap elemen matriks tersebut dengan determinan matriks semula.

2.3.2. Metode Eliminasi Gauss Jordan

Didefinisikan suatu matriks *A*, balikan dari matriks tersebut dapat dicari dengan melakukan eliminasi Gauss Jordan pada matriks *A* yang telah diaugmented dengan matriks identitas.

2.4. Interpolasi Polinom

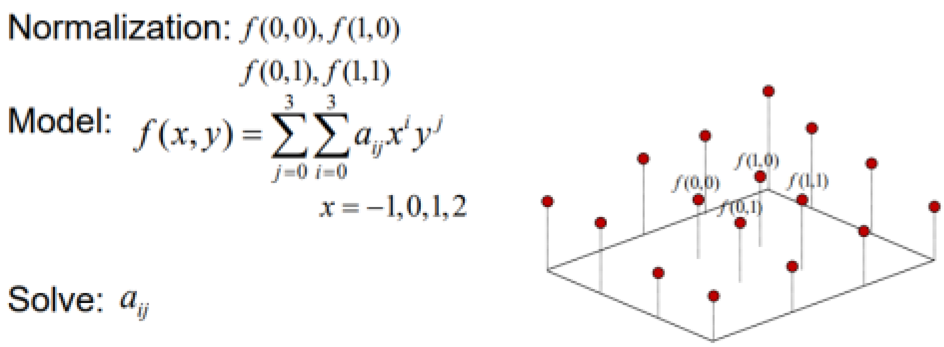
Interpolasi polinom merupakan teknik interpolasi dengan mengasumsikan pola data yang kita miliki mengikuti pola polinomial baik berderajat satu (linier) maupun berderajat tinggi. Interpolasi dengan metode ini dilakukan dengan terlebih dahulu membentuk persamaan polinomial $P_n(x)$ dari n+1 buah titik berbeda, (x_0, y_0), (x_1, y_1), …, (x_n, y_n) sehingga $y_i = P_n(x_i)$ untuk i = 0, 1, 2, …, n. Sehingga, kita dapat menggunakan persamaan polinomial tersebut untuk menghitung perkiraan nilai y di x sembarang.



2.5. Bicubic Interpolation

Bicubic interpolation merupakan teknik interpolasi pada data 2 dimensi yang merupakan pengembangan dari interpolasi cubic. Interpolasi ini dilakukan dengan mengambil data-data yang sudah diketahui untuk memprediksi nilai baru yang tidak diketahui sebelumnya.

Semisal untuk mencari nilai (x,y) yang berada pada rentang (0,0), (0,1), (1,0), hingga (1,1); kita akan memerlukan 16 titik yang sama atau bersebelahan dengan range titik yang ingin dicari. Titik-titik tersebut kemudian diplot dalam suatu fungsi berikut



2.6. Regresi Linear Berganda

Regresi linear adalah metode untuk memprediksi nilai dari suatu variabel dependen jika diberikan satu atau lebih variabel independen. Hal ini dilakukan dengan memprediksi persamaan yang berlaku menggunakan data-data yang ada hingga membentuk sebuah persamaan linear.

Rumus yang umum digunakan untuk regresi linear berganda adalah sebagai berikut

$$y_i = \beta_0 + \beta_1 x_{1i} + \beta_2 x_{2i} + \cdots + \beta_k x_{ki} + \epsilon_i$$

dimana
y = variabel dependen yang akan ditentukan nilainya
β = koefisien regresi
x = variabel independen

BAB 3
IMPLEMENTASI PUSTAKA DAN PROGRAM DALAM JAVA

1. **Folder lib (berisi pustaka yang digunakan dalam program utama)**

1.1. Matrix.java

Class ini digunakan untuk mengimplementasikan matriks yang akan digunakan di program utama.

- **Atribut**

Atribut	Deskripsi
`private int row`	Jumlah baris yang digunakan pada matriks berupa integer
`private int col`	Jumlah kolom yang digunakan pada matriks berupa integer
`private double[][] Matrix`	Elemen-elemen dari matriks berupa double

- **Konstruktor**

Konstruktor	Deskripsi
`public Matrix(int row, int col)`	Dibuat matriks dengan mengisi atribut row = row dan mengisi atribut col = col, dan menginisiasi array static berukuran row x col

- **Method**

Method	Deskripsi
`public double getElmt(int i, int j)`	Mengembalikan nilai elemen pada indeks posisi (i, j)
`public int getRowEff()`	Mengembalikan jumlah baris matriks
`public int getColEff()`	Mengembalikan jumlah kolom matriks
`public void setElmt(int i,`	Mengubah nilai elemen pada

`int j, double x)`	indeks posisi (i, j) dengan nilai x
`public static Matrix inputMatrix()`	Melakukan prosedur input matriks
`public void printMatrix()`	Melakukan prosedur output matriks

1.2. SPL.java

Class ini digunakan untuk menyelesaikan suatu persoalan sistem persamaan linear.

- **Atribut**

Atribut	Deskripsi
`private double[] x`	Nilai x hasil melakukan penyelesaian SPL dengan metode yang ada (solusi tunggal)
`private String[] ans`	Output dari hasil penyelesaian SPL (solusi tunggal, banyak, atau tidak ada)
`private Integer nEff`	Jumlah elemen pada array ans

- **Konstruktor**

Konstruktor	Deskripsi
`public SPL()`	Melakukan inisiasi nilai pada atribut yang ada

- **Method**

Method	Deskripsi
`public static void DriverSPL ()`	Prosedur untuk menyelesaikan persoalan SPL, berisi input matriks dan metode yang ingin digunakan
`public void Gauss(Matrix M)`	Penyelesaian SPL dengan metode Gauss
`public void GaussJordan`	Penyelesaian SPL dengan metode

`(Matrix M)`	Gauss Jordan
`public void InversMatrix (Matrix M)`	Penyelesaian SPL dengan metode invers matriks
`public void Cramer(Matrix M)`	Penyelesaian SPL dengan kaidah cramer
`public Matrix EselonBaris (Matrix M)`	Mengubah bentuk matriks semula dengan OBE menghasilkan matriks eselon baris
`public Matrix EselonBarisTereduksi (Matrix M)`	Mengubah bentuk matriks semula dengan OBE menghasilkan matriks eselon baris tereduksi
`public void SolveManySolution (Matrix M)`	Prosedur untuk menyelesaikan SPL dengan solusi banyak

1.3. Determinant.java

Class ini digunakan untuk menyelesaikan suatu persoalan pencarian determinan.

- **Atribut**
  -
- **Konstruktor**
  -
- **Method**

Method	Deskripsi
`public static void DriverDeterminan()`	Prosedur untuk menyelesaikan persoalan SPL, input berupa matriks persegi dan metode yang ingin digunakan. Output berupa determinan matriks.
`public static double DetOBE(Matrix M)`	Pencarian determinan dengan metode reduksi baris.
`public static double DetCofactor(Matrix M)`	Pencarian determinan dengan metode ekspansi kofaktor.

1.4. Balikan.java

Class ini digunakan untuk mencari invers dari suatu matriks.

- **Atribut**
  -
- **Konstruktor**

Method	Deskripsi
`public static boolean isInversExist`	Nilai kebenaran apakah invers suatu matriks input ada

- **Method**

Method	Deskripsi
`public static void DriverBalikan()`	Prosedur untuk mencari invers dari suatu matrix. Input berupa matriks dan metode yang ingin digunakan. Output berupa invers dari matrix.
`public static Matrix swapRow(Matrix matrix, int n, int m)`	Fungsi untuk menukar row n dan m pada matrix.
`public static Matrix Adjoin(Matrix matrix)`	Fungsi untuk mencari adjoin dari matrix.
`public static Matrix BalikanAdjoin(Matrix matrix)`	Fungsi untuk mencari invers dari matrix menggunakan matrix adjoin.
`public static Matrix BalikanGaussJordan(Matrix matrix)`	Fungsi untuk mencari invers dari matrix menggunakan Gauss Jordan.

1.5. Kofaktor.java

- **Atribut**
  -
- **Konstruktor**
  -
- **Method**

Method	Deskripsi
`public static double`	Fungsi untuk mencari nilai

`findDetKofaktor(Matrix matrixKofaktor, int row, int col)`	determinan kofaktor pada indeks (row, col)
`public static Matrix Kofaktor(Matrix matrix)`	Fungsi untuk mencari kofaktor dari matrix.

2. **Interpolate.java**

Class ini digunakan untuk menyelesaikan persoalan interpolasi polinom.

- **Atribut**

  -
- **Konstruktor**

  -
- **Method**

Method	Deskripsi
`public static void SolveInterpolate()`	Prosedur untuk menyelesaikan persoalan interpolasi polinom, berisi input titik-titik dan nilai x yang ingin diperkirakan nilainya dan output berupa fungsi f(x) beserta nilainya

3. **Regresi.java**

Class ini digunakan untuk menyelesaikan persoalan regresi linear berganda.

- **Atribut**

  -
- **Konstruktor**

  -
- **Method**

Method	Deskripsi
`public static void SolveRegression()`	Prosedur untuk menyelesaikan persoalan regresi linear berganda. Input berupa titik-titik x1,x2..,xk,y dan nilai x1,x2,..,xk yang ingin diperkirakan hasilnya. Output berupa fungsi y beserta nilai

	hampirannya.

4. **Bicubic.java**

Class ini digunakan untuk menyelesaikan persoalan bicubic interpolation.

- **Atribut**

  -
- **Konstruktor**

  -
- **Method**

Method	Deskripsi
`public static void SolveBicubic()`	Prosedur untuk menyelesaikan persoalan bicubic interpolation. Input berupa masukan dari file yang berisi titik-titik f(-1,-1), f(-1,0), f(-1,1), dst. dan nilai a dan b, yaitu titik yang nilainya ingin dicari. Output berupa fungsi f(a,b) beserta nilai hampirannya.
`public static double getValue(Matrix matrix, double a, double b)`	Fungsi untuk mencari nilai hasil bicubic interpolation.

5. **ImageProcessing.java**

Class ini digunakan untuk memperbesar citra dengan bicubic interpolation.

- **Atribut**

Method	Deskripsi
`public static int[][] columnExtendMatrix`	Array 2d hasil pembesaran lebar.
`public static int[][] columnInterpolateMatrix`	Array 2d hasil interpolasi columnExtendMatrix.
`public static int[][] rowExtendMatrix`	Array 2d hasil pembesaran tinggi.
`public static int[][]`	Array 2d hasil interpolasi

`rowInterpolateMatrix`	rowExtendMatrix.
`public static int[][] interpolateMatrix`	Array 2d hasil interpolasi salah satu dimensi.
`public static Matrix xValueMatrix`	Matrix berisi nilai konstanta yang digunakan pada double cubic interpolation.
`public static double[][] valueXandY`	Array 2d berisi nilai konstanta yang digunakan pada bicubic interpolation.

- **Konstruktor**
  -
- **Method**

Method	Deskripsi
`public static void ImageProcessingDriver()`	Prosedur untuk memperbesar citra dengan menggunakan bicubic interpolation atau double cubic interpolation. Input berupa masukan nama file pada folder test/image/, nama file hasil output image, faktor pembesaran lebar, dan faktor pembesaran tinggi. Output berupa file dengan nama yang telah diinput.
`public static void bicubic(String readDir, String writeDir, int heightFactor, int widthFactor)`	Prosedur untuk memperbesar citra dengan menggunakan bicubic interpolation.
`public static double bicubicInterpolation(int[] zValue, double a, double b)`	Fungsi untuk menghitung nilai hasil bicubic interpolation.
`public static int checkValue(double val)`	Fungsi untuk mengecek apakah nilai val melebihi 0xff atau kurang dari 0x0.

`public static void doubleCubic(String readDir, String writeDir, int heightFactor, int widthFactor)`	Prosedur untuk memperbesar citra dengan menggunakan cubic interpolation dua kali.
`public static int[][] interpolatePoints(int width, int height, int factor, boolean interpolateHeight)`	Fungsi untuk menghitung nilai elemen matriks hasil perbesaran interpolasi cubic.
`public static int getValue(int index, double x, boolean interpolateHeight)`	Fungsi untuk menghitung nilai ARGB hasil interpolasi.
`public static int RGBValue(int[] ordinate, double x)`	Fungsi untuk menghitung nilai pada titik f(x).

6. **Main.java**

Class ini digunakan sebagai penyatu semua program.

- **Atribut**

  -
- **Konstruktor**

  -
- **Method**

Method	Deskripsi
`public static void main(String[] args)`	Prosedur utama untuk menjalankan program.

BAB 4
EKSPERIMEN

4.1. Temukan solusi SPL $Ax = b$, berikut:

a. Test case 1

$$A = \begin{bmatrix} 1 & 1 & -1 & -1 \\ 2 & 5 & -7 & -5 \\ 2 & -1 & 1 & 3 \\ 5 & 2 & -4 & 2 \end{bmatrix}, \quad b = \begin{bmatrix} 1 \\ -2 \\ 4 \\ 6 \end{bmatrix}$$

- Metode Gauss:

```
Solusi tidak ada!
```

- Metode Gauss Jordan:

```
Solusi tidak ada!
```

- Metode Matriks Balikan:

```
Tidak dapat menggunakan metode matriks balikan!
```

- Kaidah Cramer:

```
Tidak dapat menggunakan kaidah cramer!
```

b. Test case 2

$$A = \begin{bmatrix} 1 & -1 & 0 & 0 & 1 \\ 1 & 1 & 0 & -3 & 0 \\ 2 & -1 & 0 & 1 & -1 \\ -1 & 2 & 0 & -2 & -1 \end{bmatrix}, \quad b = \begin{bmatrix} 3 \\ 6 \\ 5 \\ -1 \end{bmatrix}$$

- Metode Gauss:

```
X1 = 3.0 + r
X2 = 2.0r
X3 = s
X4 = -1.0 + r
X5 = r
```

- Metode Gauss Jordan:

```
X1 = 3.0 + r
X2 = 2.0r
X3 = s
X4 = -1.0 + r
X5 = r
```

- Metode Matriks Balikan:

```
Tidak dapat menggunakan metode matriks balikan!
```

- Kaidah Cramer:

```
Tidak dapat menggunakan kaidah cramer!
```

c. Test case 3

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}, \quad b = \begin{bmatrix} 2 \\ -1 \\ 1 \end{bmatrix}$$

- Metode Gauss:

```
X1 = r
X2 = 1.0 - s
X3 = t
X4 = -2.0 - s
X5 = 1.0 + s
X6 = s
```

- Metode Gauss Jordan:

```
X1 = r
X2 = 1.0 - s
X3 = t
X4 = -2.0 - s
X5 = 1.0 + s
X6 = s
```

- Metode Matriks Balikan:

```
Tidak dapat menggunakan metode matriks balikan!
```

- Kaidah Cramer:

```
Tidak dapat menggunakan kaidah cramer!
```

d. Test case 4

$$H = \begin{bmatrix} 1 & \frac{1}{2} & \frac{1}{3} & \cdots & \frac{1}{n} \\ \frac{1}{2} & \frac{1}{3} & \frac{1}{4} & \cdots & \frac{1}{n+1} \\ \frac{1}{3} & \frac{1}{4} & \frac{1}{5} & \cdots & \frac{1}{n+2} \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ \frac{1}{n} & \frac{1}{n+1} & \frac{1}{n+2} & \cdots & \frac{1}{2n+1} \end{bmatrix}, \quad b = \begin{bmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

Kasus n = 6:

- Metode Gauss:

```
X1 = 36.00000000098032
X2 = -630.0000000292666
X3 = 3360.000000203484
X4 = -7560.000000539232
X5 = 7560.000000603351
X6 = -2772.000000240222
```

- Metode Gauss Jordan:

```
X1 = 36.00000000098021
X2 = -630.0000000292657
X3 = 3360.000000203484
X4 = -7560.000000539233
X5 = 7560.000000603351
X6 = -2772.000000240222
```

- Metode Matriks Balikan:

```
X1 = 36.00000081565133
X2 = -629.9999903548862
X3 = 3359.9999267561557
X4 = -7559.999898835733
X5 = 7560.000003320784
X6 = -2772.0000426857528
```

- Kaidah Cramer:

```
X1 = 35.99999849275147
X2 = -629.9999497041402
X3 = 3359.9997099521784
X4 = -7559.999411026779
X5 = 7559.999515511824
X6 = -2771.9998638224647
```

Kasus n = 10:

- Metode Gauss:

```
Solusi tidak ada!
```

- Metode Gauss Jordan:

```
X1 = 61.9411774440141 + 1.5365259724701985E-4r
X2 = -1884.705931937533 - 0.010537906079486875r
X3 = 18099.530027735702 + 0.17391537133113144r
X4 = -78712.9443290779 - 1.1809229519294666r
X5 = 174880.59612710495 + 3.9602077873341326r
X6 = -198605.65700393674 - 6.8315679708074r
X7 = 95389.41725007199 + 5.3044984148463r
X8 = 3633.882089358747 - 0.03321801983535977r
X9 = -12870.000583849733 - 2.3823529473618588r
X10 = r
```

- Metode Matriks Balikan:

```
X1 = 5.593269546300311E12
X2 = -6.16954752068746E13
X3 = 6.07385335946012 4E10
X4 = 9.21412116466053 4E14
X5 = -1.521937691871841 8E15
X6 = -5.136719166740288E14
X7 = 1.225307362898964 5E15
X8 = 8.721926412047642E14
X9 = -4.538480853389781E14
X10 = -4.859800575577952 5E14
```

- Kaidah Cramer:

```
X1 = 34.59017767772993
X2 = -381.54024790908835
X3 = 0.37562228166023426
X4 = 5698.242961320487
X5 = -9412.043303205875
X6 = -3176.6755953264324
X7 = 7577.607165479928
X8 = 5393.857417158976
X9 = -2806.7100611943056
X10 = -3005.422213180605
```

4.2. SPL berbentuk matriks augmented

a. Test case 1

$$\begin{bmatrix} 1 & -1 & 2 & -1 & -1 \\ 2 & 1 & -2 & -2 & -2 \\ -1 & 2 & -4 & 1 & 1 \\ 3 & 0 & 0 & -3 & -3 \end{bmatrix}.$$

- Metode Gauss:

```
X1 = -1.0 + r
X2 = 2.0s
X3 = s
X4 = r
```

- Metode Gauss Jordan:

```
X1 = -1.0 + r
X2 = 2.0s
X3 = s
X4 = r
```

- Metode Matriks Balikan:

```
Tidak dapat menggunakan metode matriks balikan!
```

- Kaidah Cramer:

```
Tidak dapat menggunakan kaidah cramer!
```

b. Test case 2

$$\begin{bmatrix} 2 & 0 & 8 & 0 & 8 \\ 0 & 1 & 0 & 4 & 6 \\ -4 & 0 & 6 & 0 & 6 \\ 0 & -2 & 0 & 3 & -1 \\ 2 & 0 & -4 & 0 & -4 \\ 0 & 1 & 0 & -2 & 0 \end{bmatrix}.$$

- Metode Gauss:

```
X1 = 0.0
X2 = 2.0
X3 = 1.0
X4 = 1.0
```

- Metode Gauss Jordan:

```
X1 = 0.0
X2 = 2.0
X3 = 1.0
X4 = 1.0
```

- Metode Matriks Balikan:

```
Tidak dapat menggunakan metode matriks balikan!
```

- Kaidah Cramer:

```
Tidak dapat menggunakan kaidah cramer!
```

4.3. SPL berbentuk

a. Test case 1

$$\begin{aligned} 8x_1 + x_2 + 3x_3 + 2x_4 &= 0 \\ 2x_1 + 9x_2 - x_3 - 2x_4 &= 1 \\ x_1 + 3x_2 + 2x_3 - x_4 &= 2 \\ x_1 + 6x_3 + 4x_4 &= 3 \end{aligned}$$

- Metode Gauss:

```
X1 = -0.2243243243243243
X2 = 0.1824324324324246
X3 = 0.7094594594594594
X4 = -0.2581081081081797
```

- Metode Gauss Jordan:

```
X1 = -0.2243243243243243
X2 = 0.1824324324324246
X3 = 0.7094594594594594
X4 = -0.25810810810810797
```

- Metode Matriks Balikan:

```
X1 = -0.22432432432432434
X2 = 0.1824324324324243
X3 = 0.7094594594594594
X4 = -0.25810810810810814
```

- Kaidah Cramer:

```
X1 = -0.22432432432432434
X2 = 0.1824324324324243
X3 = 0.7094594594594594
X4 = -0.2581081081081081
```

b. Test case 2

$$
\begin{aligned}
x_7 + x_8 + x_9 &= 13.00 \\
x_4 + x_5 + x_6 &= 15.00 \\
x_1 + x_2 + x_3 &= 8.00 \\
0.04289(x_3 + x_5 + x_7) + 0.75(x_6 + x_8) + 0.61396x_9 &= 14.79 \\
0.91421(x_3 + x_5 + x_7) + 0.25(x_2 + x_4 + x_6 + x_8) &= 14.31 \\
0.04289(x_3 + x_5 + x_7) + 0.75(x_2 + x_4) + 0.61396x_1 &= 3.81 \\
x_3 + x_6 + x_9 &= 18.00 \\
x_2 + x_5 + x_8 &= 12.00 \\
x_1 + x_4 + x_7 &= 6.00 \\
0.04289(x_1 + x_5 + x_9) + 0.75(x_2 + x_6) + 0.61396x_3 &= 10.51 \\
0.91421(x_1 + x_5 + x_9) + 0.25(x_2 + x_4 + x_6 + x_8) &= 16.13 \\
0.04289(x_1 + x_5 + x_9) + 0.75(x_4 + x_8) + 0.61396x_7 &= 7.04
\end{aligned}
$$

- Metode Gauss:

```
Solusi tidak ada!
```

- Metode Gauss Jordan:

```
Solusi tidak ada!
```

- Metode Matriks Balikan:

```
Tidak dapat menggunakan metode matriks balikan!
```

- Kaidah Cramer:

```
Tidak dapat menggunakan kaidah cramer!
```

4.4. Studi Kasus Interpolasi

a. Test case 1

Gunakan tabel di bawah ini untuk mencari polinom interpolasi dari pasangan titik-titik yang terdapat dalam tabel. Program menerima masukan nilai x yang akan dicari nilai fungsi $f(x)$.

x	0.4	0.7	0.11	0.14	0.17	0.2	0.23
$f(x)$	0.043	0.005	0. 058	0.072	0.1	0.13	0.147

Lakukan pengujian pada nilai-nilai default berikut:

- x = 0.2, f(x) = ?

```
f(x) = -4212.434532x^6 + 7102.399163x^5 - 4346.313951x^4 + 1220.854891x^3 - 163.915663x^2 + 10.276384x - 0.184559,
f(0.200000) = 0.130000
```

- x = 0.55, f(x) = ?

```
f(x) = -4212.434532x^6 + 7102.399163x^5 - 4346.313951x^4 + 1220.854891x^3 - 163.915663x^2 + 10.276384x - 0.184559,
f(0.550000) = 2.137572
```

- x = 0.85, f(x) = ?

```
f(x) = -4212.434532x^6 + 7102.399163x^5 - 4346.313951x^4 + 1220.854891x^3 - 163.915663x^2 + 10.276384x - 0.184559,
f(0.850000) = -66.269639
```

- x = 1.28, f(x) = ?

```
f(x) = -4212.434532x^6 + 7102.399163x^5 - 4346.313951x^4 + 1220.854891x^3 - 163.915663x^2 + 10.276384x - 0.184559,
f(1.280000) = -3485.144902
```

b. Test case 2

Jumlah kasus positif baru Covid-19 di Indonesia semakin fluktuatif dari hari ke hari. Di bawah ini diperlihatkan jumlah kasus baru Covid-19 di Indonesia mulai dari tanggal 17 Juni 2022 hingga 31 Agustus 2022:

Tanggal	Tanggal (desimal)	Jumlah Kasus Baru
17/06/2022	6,567	12.624
30/06/2022	7	21.807
08/07/2022	7,258	38.391
14/07/2022	7,451	54.517
17/07/2022	7,548	51.952
26/07/2022	7,839	28.228
05/08/2022	8,161	35.764
15/08/2022	8,484	20.813
22/08/2022	8,709	12.408
31/08/2022	9	10.534

Tanggal (desimal) adalah tanggal yang sudah diolah ke dalam bentuk desimal 3 angka di belakang koma dengan memanfaatkan perhitungan sebagai berikut:

$$\boxed{\text{tanggal(desimal)} = \text{bulan} + (\text{tanggal} / \text{jumlah hari pada bulan tersebut})}$$

Sebagai **contoh**, untuk tanggal 17/06/2022 (dibaca: 17 Juni 2022) diperoleh tanggal(desimal) sebagai berikut:

$$\text{Tanggal(desimal)} = 6 + (17/30) = 6{,}567$$

Gunakanlah data di atas dengan memanfaatkan **polinom interpolasi** untuk melakukan prediksi jumlah kasus baru Covid-19 pada tanggal-tanggal berikut:

- 16/07/2022 (7.516)

```
f(x) = -141120.310600x^9 + 9381759.266086x^8 - 275752903.603842x^7 + 4700873047.890322x^6 - 51191089915.822266x^5 + 369011568500.298100x^4 - 1759197443156.986300x^3
 + 5342144345318.836000x^2 - 9362383549278.719000x + 7200305831156.062500,
f(7.516000) = 53537.855469
```

- 10/08/2022 (8.323)

```
f(x) = -141120.310600x^9 + 9381759.266086x^8 - 275752903.603842x^7 + 4700873047.890322x^6 - 51191089915.822266x^5 + 369011568500.298100x^4 - 1759197443156.986300x^3
 + 5342144345318.836000x^2 - 9362383549278.719000x + 7200305831156.062500,
f(8.323000) = 36295.816406
```

- 05/09/2022 (9.167)

```
f(x) = -141120.310600x^9 + 9381759.266086x^8 - 275752903.603842x^7 + 4700873047.890322x^6 - 51191089915.822266x^5 + 369011568500.298100x^4 - 1759197443156.986300x^3
 + 5342144345318.836000x^2 - 9362383549278.719000x + 7200305831156.062500,
f(9.167000) = -667870.812500
```

- 03/10/2022 (10.097)

```
f(x) = -141120.310600x^9 + 9381759.266086x^8 - 275752903.603842x^7 + 4700873047.890322x^6 - 51191089915.822266x^5 + 369011568500.298100x^4 - 1759197443156.986300x^3
 + 5342144345318.836000x^2 - 9362383549278.719000x + 7200305831156.062500,
f(10.097000) = -332900836.562500
```

c. Test case 3

Sederhanakan fungsi

$$f(x) = \frac{x^2 + \sqrt{x}}{e^x + x}$$

dengan polinom interpolasi derajat n di dalam selang [0, 2]. Sebagai contoh, jika n = 5, maka titik-titik x yang diambil di dalam selang [0, 2] berjarak $h = (2 – 0)/5 = 0.4$.

- Untuk n = 5, titik-titik x yang diambil = {0, 0.4, 0.8, 1.2, 1.6, 2.0}

```
f(x) = 0.236256x^5 - 1.421265x^4 + 3.237114x^3 - 3.552684x^2 + 2.035259x + 0.000000
```

4.5 Studi Kasus Interpolasi Bicubic

a. TC-Bicubic-1.txt

```
f(0.0,0.0) = 161.0
```

b. TC-Bicubic-2.txt

```
f(0.5,0.5) = 97.7265625
```

c. TC-Bicubic-3.txt

```
f(0.25,0.75) = 105.51477050781251
```

d. TC-Bicubic-4.txt

```
f(0.1,0.9) = 104.22911850000003
```

4.6 Studi Kasus Pembesaran Citra dengan Interpolasi Bicubic

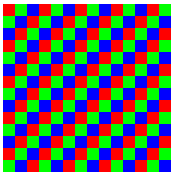
a. TC-ImageProcessing-1.jpg

Faktor pembesaran : 2

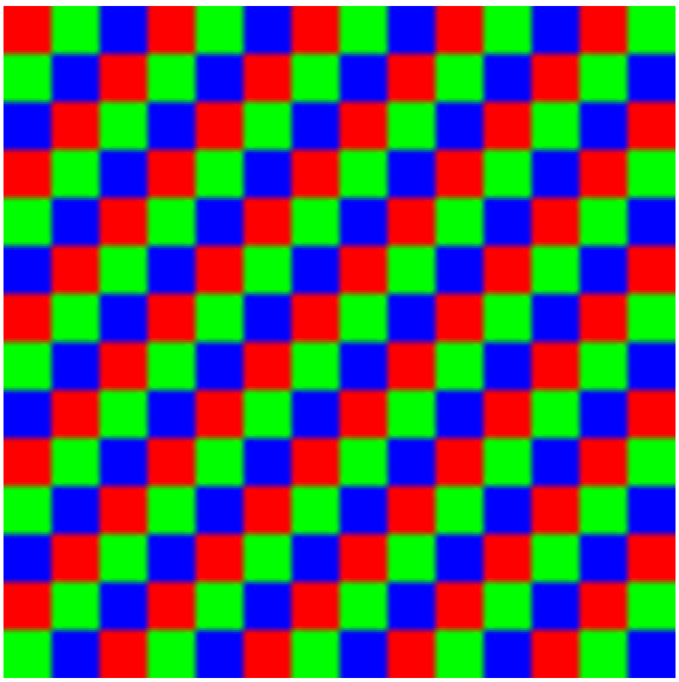
Input :

Output :

b. TC-ImageProcessing-2.png
Faktor pembesaran : 4
Input :



Output :

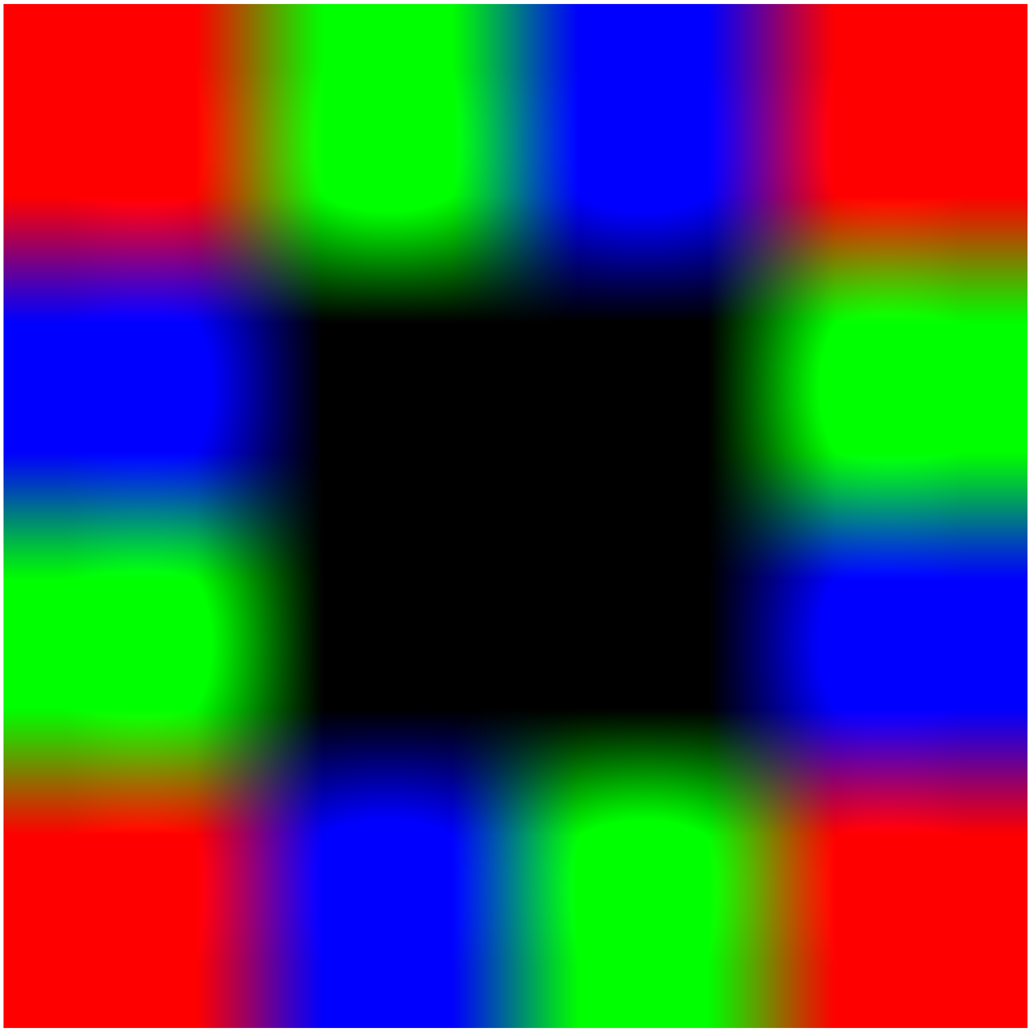


c. TC-ImageProcessing-3.jpg
Faktor pembesaran : 3
Input :

Output :

d. TC-ImageProcessing-4.png
   Faktor pembesaran : 64
   Input :

   Output :



4.7 Studi Kasus Linear Berganda

Diberikan sekumpulan data sesuai pada tabel berikut ini.

Table 12.1: Data for Example 12.1

Nitrous Oxide, y	Humidity, x_1	Temp., x_2	Pressure, x_3	Nitrous Oxide, y	Humidity, x_1	Temp., x_2	Pressure, x_3
0.90	72.4	76.3	29.18	1.07	23.2	76.8	29.38
0.91	41.6	70.3	29.35	0.94	47.4	86.6	29.35
0.96	34.3	77.1	29.24	1.10	31.5	76.9	29.63
0.89	35.1	68.0	29.27	1.10	10.6	86.3	29.56
1.00	10.7	79.0	29.78	1.10	11.2	86.0	29.48
1.10	12.9	67.4	29.39	0.91	73.3	76.3	29.40
1.15	8.3	66.8	29.69	0.87	75.4	77.9	29.28
1.03	20.1	76.9	29.48	0.78	96.6	78.7	29.29
0.77	72.2	77.7	29.09	0.82	107.4	86.8	29.03
1.07	24.0	67.7	29.60	0.95	54.9	70.9	29.37

Source: Charles T. Hare, "Light-Duty Diesel Emission Correction Factors for Ambient Conditions," EPA-600/2-77-116. U.S. Environmental Protection Agency.

Gunakan *Normal Estimation Equation for Multiple Linear Regression* untuk mendapatkan regresi linear berganda dari data pada tabel di atas, kemudian estimasi nilai Nitrous Oxide apabila Humidity bernilai 50%, temperatur 76°F, dan tekanan udara sebesar 29.30.

Persamaan yang didapatkan:

```
Persamaannya adalah
y =  -3.5077781408835103 -0.002624990745878327 x1 +7.989410472218274E-4 x2 +0.15415503019830143 x3
```

Hampiran nilai y dari data yang diinginkan:

```
Hampiran nilai y-nya adalah y=0.9384342262216645
```

4.8. Studi kasus pencarian determinan

a. Test case 1

```
153 59 210 96
125 161 72 81
98 101 42 12
21 51 0 16
```

-Metode reduksi baris

```
Determinan dari matriks input adalah:
6781745.99999999
```

-Metode ekspansi kofaktor

```
Determinan dari matriks input adalah:
6781746.0
```

b. Test case 2

```
0 0 0 1
0 0 1 0
0 2 0 0
4 0 0 0
```

-Metode reduksi baris

```
Determinan dari matriks input adalah:
8.0
```

-Metode ekspansi kofaktor

```
Determinan dari matriks input adalah:
8.0
```

BAB 5
KESIMPULAN

5.1. Kesimpulan

Terdapat berbagai metode dalam penyelesaian SPL, pencarian determinan, dan penentuan matriks balikan. Cara penyelesaian SPL di antaranya adalah menggunakan metode eliminasi Gauss, metode Gauss-Jordan, metode matriks balikan, dan kaidah Cramer. Cara pencarian determinan di antaranya adalah menggunakan metode reduksi baris dan metode ekspansi kofaktor.

Basis-basis di atas ternyata dapat menyelesaikan berbagai masalah dengan pengaplikasian yang sesuai. Beberapa di antara pengaplikasiannya adalah ekspansi bikubik, interpolasi polinom, dan juga regresi linear berganda.

Pada tugas ini, kami mengimplementasi sebuah program yang dapat mengimplementasi metode-metode dan pengaplikasiannya yang telah disebutkan di atas. Program ini diimplementasi dalam bahasa Java dan dapat menyelesaikan berbagai permasalahan yang melibatkan matriks, seperti penggunaan algoritma ekspansi bikubik untuk memperbesar ukuran gambar.

5.2. Saran

Beberapa saran pengembangan dari program ini adalah sebagai berikut:

- Penggunaan *rounding* untuk menghasilkan hasil kalkulasi yang lebih akurat.
- Penggunaan metode pencarian determinan yang lebih beragam.
- Penjabaran *step-by-step* dari metode yang digunakan pada penyelesaian SPL atau pencarian determinan sebagai output apabila diminta.

5.3. Refleksi

Pengerjaan tubes ini tergolong cukup lancar karena progresinya yang cukup konstan per harinya dan pembagian tugas yang sudah cukup jelas. Komunikasi yang konstan juga menjamin pengerjaan tubes tidak stagnan. Selain itu, dibutuhkan ketelitian lebih dalam membaca instruksi yang telah diberikan agar tidak perlu banyak *adjustment* pada program yang telah dibuat.

DAFTAR PUSTAKA

Informatika.stei.itb.ac.id. (2022). Determinan (Bagian 1). Diakses pada 28 September 2022, dari https://informatika.stei.itb.ac.id/~rinaldi.munir/AljabarGeometri/2020-2021/Algeo-08-Determinan-bagian1.pdf

Informatika.stei.itb.ac.id. (2022). Determinan (Bagian 2). Diakses pada 28 September 2022, dari https://informatika.stei.itb.ac.id/~rinaldi.munir/AljabarGeometri/2020-2021/Algeo-09-Determinan-bagian2.pdf

Informatika.stei.itb.ac.id. (2022). Sistem persamaan linier (Bagian 1: Metode eliminasi Gauss). Diakses pada 20 September 2022, dari https://informatika.stei.itb.ac.id/~rinaldi.munir/AljabarGeometri/2020-2021/Algeo-03-Sistem-Persamaan-Linier.pdf

Informatika.stei.itb.ac.id. (2022). Sistem persamaan linier (Bagian 2: Metode eliminasi Gauss-Jordan). Diakses pada 22 September 2022, dari https://informatika.stei.itb.ac.id/~rinaldi.munir/AljabarGeometri/2020-2021/Algeo-05-Sistem-Persamaan-Linier-2.pdf

Link repository: https://github.com/briankheng/Algeo01-21049