

LAPORAN TUGAS BESAR 1
IF2123 - ALJABAR LINIER DAN GEOMETRI



Kelompok 23 “Opet Brot Brot”

Shazya Audrea Taufik 13522063

Marzuli Suhada M 13522070

William Glory Henderson 13522113

SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA
INSTITUT TEKNOLOGI BANDUNG

2023

BAB I

DESKRIPSI MASALAH

Sistem persamaan linier (SPL) banyak ditemukan di dalam bidang sains dan rekayasa. Anda sudah mempelajari berbagai metode untuk menyelesaikan SPL, termasuk menghitung determinan matriks. 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).

SPL, determinan, dan invers dapat digunakan untuk menyelesaikan permasalahan/persoalan interpolasi dan regresi. Dalam tugas besar ini, perlu dibuat sebuah library yang berisi fungsi-fungsi seperti eliminasi Gauss, eliminasi Gauss-Jordan, menentukan balikan matriks, menghitung determinan, kaidah Cramer (kaidah Cramer khusus untuk SPL dengan n peubah dan n persamaan). Selanjutnya, gunakan *library* tersebut di dalam program Java untuk menyelesaikan berbagai persoalan yang dimodelkan dalam bentuk SPL, menyelesaikan persoalan interpolasi, dan persoalan regresi.

BAB II

TEORI SINGKAT

2.1 Sistem Persamaan Linier

2.1.1 Metode Eliminasi Gauss

Metode Eliminasi Gauss adalah salah satu teknik yang dapat dipakai dalam menyelesaikan persoalan Solusi Persamaan Linear dalam bentuk matriks. Cara kerja metode eliminasi Gauss ini adalah dengan mengubah bentuk matriksnya menjadi lebih sederhana yaitu dengan membuat elemen tak 0 paling kiri dalam tiap barisnya bernilai 1. Nilai ini disebut sebagai 1 utama. OBE dapat diterapkan dalam penyelesaian ini. Selanjutnya, elemen-elemen yang berada dibawah 1 utama pada tiap kolom dibuat menjadi 0. Setelah matriksnya dibuat, penyelesaian sistem solusi persamaan linearnya menggunakan substitusi mundur.

$$\left[\begin{array}{ccccc} a_{11} & a_{12} & \dots & a_{1n} & b_1 \\ a_{21} & a_{22} & \dots & a_{2n} & b_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} & b_n \end{array} \right] \sim_{\text{OBE}} \left[\begin{array}{cccccc} 1 & * & * & \dots & * & * \\ 0 & 1 & * & \dots & * & * \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \vdots & 1 & * \end{array} \right]$$

2.1.2 Metode Eliminasi Gauss-Jordan

Metode Eliminasi Gauss-Jordan adalah salah satu teknik yang juga dapat dipakai dalam menyelesaikan persoalan Solusi Persamaan Linear dalam bentuk matriks. Sama seperti metode eliminasi gauss jordan namun perbedaannya adalah tidak hanya elemen-elemen yang berada dibawah 1 utama saja yang dibuat 0, namun juga elemen-elemen yang berada diatas 1 utama dibuat menjadi 0 menggunakan OBE. Sehingga penyelesaian SPL-nya nanti tidak perlu lagi melakukan substitusi mundur, cukup disamakan dengan elemen-elemen pada kolom terakhir saja.

$$\left[\begin{array}{ccccc} a_{11} & a_{12} & \dots & a_{1n} & b_1 \\ a_{21} & a_{22} & \dots & a_{2n} & b_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} & b_m \end{array} \right] \sim_{\text{OBE}} \left[\begin{array}{cccccc} 1 & 0 & 0 & \dots & 0 & * \\ 0 & 1 & 0 & \dots & 0 & * \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \vdots & 1 & * \end{array} \right]$$

2.1.3 Metode Matriks Balikan

Penyelesaian Sistem Persamaan Linear menggunakan metode matriks balikan adalah sebuah pendekatan matematika yang digunakan untuk menemukan solusi dari rangkaian persamaan linear $Ax = b$, di mana A adalah matriks koefisien, x adalah solusi yang ingin dicari, dan b adalah konstanta. Metode ini melibatkan konsep matriks augmented $[A | b]$, yang menggabungkan matriks koefisien A dengan vektor konstanta b. Kunci dari metode ini adalah adanya matriks balikan A^{-1} , yang merupakan invers dari matriks A. Dengan kata lain, A^{-1} memenuhi persamaan $A^{-1}A = I$, di mana I adalah matriks identitas. Dengan memanfaatkan matriks balikan, solusi dari sistem persamaan linear dapat ditemukan dengan mudah menggunakan rumus $x = A^{-1}b$. Namun metode matriks balikan tidak dapat menyelesaikan SPL yang memiliki jumlah variabel dan persamaan yang besar.

2.1.4 Kaidah Cramer

Kaidah Cramer juga menjadi salah satu alternatif metode dalam menyelesaikan sistem persamaan linear. Penyelesaian SPL menggunakan kaidah Cramer adalah dengan mencari nilai determinan dari matriks koefisien dan determinan matriks yang dihasilkan dengan menggantikan setiap kolom matriks koefisien A dengan konstanta b pada $Ax = b$. Untuk setiap variabel dalam SPL, kaidah Cramer menghitung solusinya secara terpisah. Namun, metode ini memiliki beberapa pembatasan penting. Pertama, SPL harus memiliki jumlah persamaan yang sama dengan jumlah variabel, sehingga matriks A adalah matriks persegi. Kedua, determinan matriks A harus tidak nol; jika determinannya nol, SPL tidak memiliki solusi unik. Dan sama dengan metode matriks balikan, metode Cramer cenderung tidak efisien untuk SPL yang besar, karena menghitung determinan matriks merupakan tugas yang kompleks dan memakan waktu.

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

2.2 Determinan

2.2.1 Metode OBE

Metode Operasi Baris Elementer (OBE) merupakan salah satu metode yang dapat digunakan untuk menghitung determinan dari sebuah matriks. Metode ini menggunakan pendekatan matematis untuk mengubah matriks awal menjadi bentuk segitiga atau bentuk

diagonal dengan mengaplikasikan operasi baris elementer seperti pertukaran baris, perkalian baris dengan suatu konstanta, atau penambahan/subtraksi baris. Ketika matriks mencapai bentuk segitiga atas atau diagonal, determinan mudah dihitung dengan mengalikan elemen-elemen diagonal utama (elemen-elemen pada baris dan kolom yang sama).

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

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

Sehingga determinannya

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

Dengan p merupakan banyaknya pertukaran baris yang dilakukan dalam OBE.

2.2.2 Metode Kofaktor

Metode kedua yang dapat digunakan untuk menghitung determinan matriks adalah menggunakan metode kofaktor, yang didasarkan pada konsep kofaktor dan aturan ekspansi kofaktor. Pertama, kita mengidentifikasi elemen mana yang akan diambil sebagai elemen utama dalam ekspansi kofaktor, biasanya dari satu baris atau satu kolom tertentu. Kemudian, untuk elemen utama tersebut, kita menghitung kofaktor yang merupakan determinan matriks yang diperoleh dengan menghapus baris dan kolom yang mengandung elemen tersebut. Setelah itu, kita mengalikan elemen utama dengan kofaktor yang sesuai, dengan mempertimbangkan tanda positif atau negatif sesuai dengan posisi elemen dalam matriks. Langkah ini diulang untuk semua elemen dalam satu baris atau satu kolom yang dipilih sebelumnya. Terakhir, menjumlahkan semua produk tersebut untuk mendapatkan nilai determinan matriks asal.

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

- Didefinisikan:
- M_{ij} = minor entri a_{ij}
= determinan upa-matriks (*submatrix*) yang elemen-elemennya tidak berada pada baris i dan kolom j
- $C_{ij} = (-1)^{i+j} M_{ij}$ = kofaktor entri a_{ij}

Dengan menggunakan kofaktor, maka determinan matriks dapat dihitung dengan salah satu dari persamaan berikut.

$$\begin{aligned} \det(A) &= a_{11}C_{11} + a_{12}C_{12} + \dots + a_{1n}C_{1n} \\ \det(A) &= a_{21}C_{21} + a_{22}C_{22} + \dots + a_{2n}C_{2n} \\ &\vdots \\ \det(A) &= a_{n1}C_{n1} + a_{n2}C_{n2} + \dots + a_{nn}C_{nn} \end{aligned}$$

Secara baris

$$\begin{aligned} \det(A) &= a_{11}C_{11} + a_{21}C_{21} + \dots + a_{n1}C_{n1} \\ \det(A) &= a_{12}C_{12} + a_{22}C_{22} + \dots + a_{n2}C_{n2} \\ &\vdots \\ \det(A) &= a_{1n}C_{1n} + a_{2n}C_{2n} + \dots + a_{nn}C_{nn} \end{aligned}$$

Secara kolom

2.3 Invers Matriks/Matriks Balikan

Matriks balikan (*inverse matrix*) adalah matriks yang, ketika dikalikan dengan matriks asalnya, menghasilkan matriks identitas. Matriks balikan sering digunakan dalam berbagai aplikasi matematika, seperti penyelesaian sistem persamaan linear dan transformasi linier. Terdapat dua metode utama untuk menghitung matriks balikan: metode adjoint dan metode eliminasi Gauss-Jordan (OBE).

2.3.1 Metode Adjoint

Metode pertama adalah metode adjoint yaitu dengan memanfaatkan kofaktor dan matriks adjoint untuk menghitung matriks balikan. Pertama, kita menghitung determinan matriks asal. Selanjutnya, kita mencari matriks adjoint, yang diperoleh dengan melakukan transpose matriks kofaktor. Kemudian, kita mengalikan matriks adjoint dengan invers dari determinan matriks asal. Hasilnya adalah matriks balikan.

$$A^{-1} = \frac{1}{\det(A)} \text{adj}(A)$$

2.3.2 Metode OBE (Identitas)

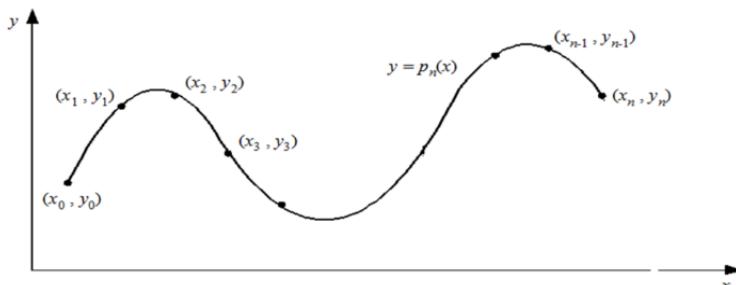
Metode yang kedua adalah dengan memanfaatkan matriks identitas yaitu mengubah matriks asal menjadi matriks identitas melalui serangkaian operasi baris elementer. Pertama, matriks asal digabungkan dengan matriks identitas (matriks augmented) menjadi matriks yang memiliki bentuk segitiga atas. Selanjutnya, matriks tersebut dibawa ke bentuk matriks

identitas dengan mengaplikasikan operasi baris elementer yang sesuai. Matriks yang dihasilkan adalah matriks balikan.

$$\begin{matrix} \text{G-J} \\ [A|I] \sim [I|A^{-1}] \end{matrix}$$

2.4 Interpolasi Polinom

Interpolasi Polinom adalah metode matematis yang digunakan untuk menemukan polinom $P_n(X)$ yang melewati sejumlah titik atau data yang diketahui, dengan $P_n(X_i) = Y_i$ untuk setiap data (x_i, y_i) . Polinom ini digunakan juga untuk memperkirakan nilai y di titik lain dalam selang $[x_0, x_n]$. Polinom interpolasi memiliki derajat n , di mana derajat menentukan bentuk polinom. Untuk menemukan koefisien polinom, dapat dibentuk sistem persamaan linear dengan menyulihkan titik-titik data ke dalamnya dan menggunakan metode eliminasi gauss untuk perhitungannya. Berikut ilustrasi beberapa titik yang diinterpolasi secara polinomial.



2.5 Interpolasi *Bicubic Spline*

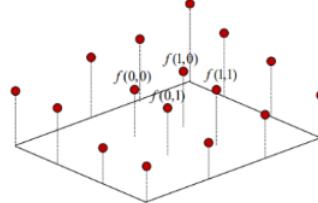
Interpolasi *Bicubic Spline* adalah metode matematis yang digunakan untuk menghubungkan titik-titik data dengan kurva yang halus. Konsepnya adalah mengambil sekelompok titik data dan membuat kurva lembut yang melewatkinya dengan menggunakan fungsi matematika khusus yang disebut "fungsi kubik." Interpolasi *Bicubic spline* membagi kurva menjadi segmen kecil dan menggunakan empat fungsi kubik untuk menggambarkan setiap segmen, dua untuk arah horizontal dan dua lagi untuk arah vertikal. Ini memungkinkan kita untuk memperkirakan nilai di antara titik-titik data yang diketahui dengan baik. Dalam Interpolasi *Bicubic Spline*, kita menggunakan 16 titik, dengan 4 titik referensi utama di tengah

dan 12 titik di sekitarnya untuk menghitung turunan dari keempat titik referensi tersebut demi membangun permukaan bicubic. Modelnya memiliki bentuk berikut:

Normalization: $f(0,0), f(1,0)$
 $f(0,1), f(1,1)$

Model: $f(x,y) = \sum_{j=0}^3 \sum_{i=0}^3 a_{ij} x^i y^j$
 $x = -1, 0, 1, 2$

Solve: a_{ij}



Dengan menggunakan nilai fungsi dan turunan berarah dari fungsi tersebut, dapat terbentuk sebuah matriks solusi X yang membentuk persamaan penyelesaian sebagai berikut.

$$\begin{bmatrix} f(0,0) \\ f(1,0) \\ f(0,1) \\ f(1,1) \\ f_x(0,0) \\ f_x(1,0) \\ f_x(0,1) \\ f_x(1,1) \\ f_y(0,0) \\ f_y(1,0) \\ f_y(0,1) \\ f_y(1,1) \\ f_{xy}(0,0) \\ f_{xy}(1,0) \\ f_{xy}(0,1) \\ f_{xy}(1,1) \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 2 & 3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 2 & 3 & 0 & 1 & 2 & 3 & 0 & 1 & 2 & 3 & 0 & 1 & 2 & 3 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 2 & 3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 2 & 3 & 0 & 2 & 4 & 6 & 0 & 3 & 6 & 9 \end{bmatrix} \begin{bmatrix} a_{00} \\ a_{10} \\ a_{20} \\ a_{30} \\ a_{01} \\ a_{11} \\ a_{21} \\ a_{31} \\ a_{02} \\ a_{12} \\ a_{22} \\ a_{32} \\ a_{03} \\ a_{13} \\ a_{23} \\ a_{33} \end{bmatrix}$$

2.6 Regresi Linier Berganda

Regresi Linier Berganda merupakan metode statistik yang digunakan untuk memodelkan hubungan yang kompleks antara satu variabel dependen dan beberapa variabel independen (peubah) menggunakan matriks. Dalam kerangka ini, data dinyatakan dalam sebuah matriks yang berisi semua variabel independen, termasuk intercept, dan vektor parameter (β) yang menyimpan koefisien regresi. Model regresi linear berganda dijelaskan sebagai persamaan matriks, dengan tujuan mengestimasi parameter β .

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

Untuk mendapatkan nilai dari setiap β_i dapat digunakan *Normal Estimation Equation for Multiple Linear Regression* sebagai berikut:

$$\begin{aligned}
 nb_0 + b_1 \sum_{i=1}^n x_{1i} + b_2 \sum_{i=1}^n x_{2i} + \cdots + b_k \sum_{i=1}^n x_{ki} &= \sum_{i=1}^n y_i \\
 b_0 \sum_{i=1}^n x_{1i} + b_1 \sum_{i=1}^n x_{1i}^2 + b_2 \sum_{i=1}^n x_{1i}x_{2i} + \cdots + b_k \sum_{i=1}^n x_{1i}x_{ki} &= \sum_{i=1}^n x_{1i}y_i \\
 \vdots &\quad \vdots \quad \vdots \quad \vdots \quad \vdots \\
 b_0 \sum_{i=1}^n x_{ki} + b_1 \sum_{i=1}^n x_{ki}x_{1i} + b_2 \sum_{i=1}^n x_{ki}x_{2i} + \cdots + b_k \sum_{i=1}^n x_{ki}^2 &= \sum_{i=1}^n x_{ki}y_i
 \end{aligned}$$

Metode eliminasi gauss dapat diterapkan untuk menyelesaikan sistem persamaan linear tersebut.

BAB III

IMPLEMENTASI PUSTAKA DAN PROGRAM DALAM JAVA

3.1 Folder ADTMatrix

3.1.1 InputMatrix.java

Class ini digunakan untuk menginput matriks dari *keyboard* dan *file*.

- **Atribut**

-

- **Konstruktor**

<code>public static Scanner input = new Scanner(System.in)</code>	Membuat scanner untuk bisa menginput.
---	---------------------------------------

- **Metode**

Nama Procedure/Function	Fungsi
<code>public static double[][] readMatrixKeyboard1()</code>	Fungsi ini menerima inputan berupa 2 matriks (matriks A dan matriks B) kemudian menggabungkan keduanya menjadi matriks augmented. Fungsi ini bisa digunakan untuk menginput matriks SPL. Fungsi ini menghasilkan matriks.
<code>public static double[][] readMatrixKeyboard2()</code>	Fungsi ini menerima inputan jumlah baris dan kolom untuk menginput matriks. Fungsi ini bisa digunakan untuk matriks augmented, determinan, dan invers. Fungsi ini menghasilkan matriks.
<code>public static double[][] readInterpolasiKeyboard()</code>	Fungsi ini menginput derajat polinom dan titik-titiknya. Dari inputan ini, akan dihasilkan matriks.
<code>public static double[][] readRegresiKeyboard()</code>	Fungsi ini menginput jumlah peubah, jumlah sampel, dan titik-titik. Dari inputan ini, akan dihasilkan matriks.
<code>public static double[][] readMatrixFile()</code>	Fungsi ini meminta inputan nama file dan membaca file tersebut. Dari file tersebut akan dihasilkan sebuah matriks.

3.1.2 Matrix.java

Class ini berisikan fungsi-fungsi ADT dan fungsi-fungsi yang digunakan untuk mempermudah akses matriks.

- **Atribut**

- **Konstruktor**

<code>public double MARK = double.NaN</code>	Inisialisasi nilai <i>mark</i> yaitu NaN (Not a Number)
<code>public Matrix (double [][] contents, int rows, int cols)</code>	Fungsi ini membuat matriks kosong dari tipe data double [][] dengan ukuran <i>rows</i> x <i>cols</i>
<code>public Matrix (int rows, int cols)</code>	Fungsi ini membuat matriks kosong dengan ukuran <i>rows</i> x <i>cols</i> .

- **Metode**

Nama Procedure/Function	Fungsi
<code>public int getRowLength()</code>	Fungsi ini mengembalikan jumlah baris
<code>public int getColLength()</code>	Fungsi ini mengembalikan jumlah kolom
<code>public int getLastRowIndex()</code>	Fungsi ini mengembalikan indeks baris terakhir
<code>public int getLastColIdx()</code>	Fungsi ini mengembalikan indeks kolom terakhir
<code>public boolean isSquare()</code>	Fungsi ini mengembalikan true jika matriks merupakan matriks persegi atau matriks nxn
<code>public double getElmt(int i, int j)</code>	Fungsi ini mengembalikan elemen matriks pada baris i dan kolom j
<code>public void setElmt(int i, int j, double elmt)</code>	Prosedur ini mengisi elemen matriks pada baris i dan kolom j dengan elemen elmt
<code>public void rowSwap(Matrix m, int rows1, int rows2)</code>	Prosedur ini menukar posisi 2 baris
<code>public static Matrix multiplyMatrix(Matrix m1,</code>	Fungsi ini mengembalikan hasil perkalian 2 matriks

Matrix m2)	
public static double sumCol(Matrix m, int i)	Fungsi ini mengembalikan nilai jumlah elemen pada suatu kolom
public static double sumMultiplyCol(Matrix m, int i, int j)	Fungsi ini mengembalikan nilai jumlah dari perkalian antara dua kolom.
public static Matrix addRow(Matrix m, double [] newRow)	Fungsi ini mengembalikan matriks yang sudah ditambahkan baris baru.
public static double detKofaktorIJ(Matrix m, int row, int col)	Fungsi ini mengembalikan nilai determinan dari dengan acuan row dan col tertentu.
public static Matrix matriksKofaktor(Matrix m)	Fungsi ini mengembalikan matriks kofaktor.
public static Matrix Adjoin (Matrix m)	Fungsi ini mengembalikan matriks adjoin atau transpose dari matriks kofaktor.
public static void BackSubstitution(Matrix, matrix, double[] X)	Prosedur mengubah matriks eselon baris menjadi matriks eselon baris tereduksi
public static int SolutionType(Matrix matrix)	Fungsi mengembalikan tipe solusi (solusi unik, solusi tidak ada, dan solusi parametrik)
public static void solveManySolution(Matrix matrix)	Prosedur menyelesaikan persamaan SPL yang memiliki banyak solusi atau parametrik
public static Matrix gaussElimination(Matrix matrix)	Fungsi mengembalikan hasil reduksi baris sehingga membentuk matriks eselon baris
public static Matrix gaussJordanElimination (Matrix A)	Fungsi mengembalikan hasil reduksi baris sehingga membentuk matriks eselon baris tereduksi

3.1.3 Main.java

- **Atribut**

-

- **Konstruktor**

-

- **Metode**

<code>public static void main(String[] args)</code>	Prosedur yang berisikan menu dan driver fungsi-fungsi. Terdapat beberapa pilihan pada menu dan sejumlah fungsi tertentu akan dijalankan jika memilih fungsi terkait.
---	--

3.1.4 OutputMatrix.java

- **Atribut**

-

- **Konstruktor**

-

- **Metode**

Nama Procedure/Function	Fungsi
<code>public static void printMatrix (Matrix matrix)</code>	Prosedur ini dapat menampilkan hasil matriks ke layar.
<code>public static int printMenuOutput()</code>	Fungsi ini menampilkan menu untuk menghasilkan file dan mengembalikan nilai pilihannya.
<code>public static void OutputFileDeterminan(double det)</code>	Prosedur ini akan membuat file dengan nama yang diinputkan jika memilih pilihan buat file output. File tersebut kemudian diisi dengan nilai determinan hasil.
<code>public static Matrix OutputFileInvers(Matrix m1)</code>	Prosedur ini akan membuat file dengan nama yang diinputkan jika memilih pilihan buat file output. File tersebut kemudian diisi dengan matriks hasil invers.

3.2 Folder Function

3.2.1 SPL.java

Class ini digunakan untuk mencari solusi sistem persamaan linier dengan empat metode, yaitu metode gauss, metode gauss-jordan, metode invers, dan kaidah cramer.

- **Atribut**

-

- **Konstruktor**

-

- **Metode**

Nama Procedure/Function	Fungsi
public static void gaussSPL(Matrix Mgauss)	Prosedur ini digunakan untuk mencari solusi SPL dengan metode gauss.
public static void gaussJordanSPL(Matrix Mgajo)	Prosedur ini digunakan untuk mencari solusi SPL dengan metode gauss-jordan.
public static void cramerSPL(Matrix mCramer)	Prosedur ini digunakan untuk mencari solusi SPL dengan metode matriks balikan.
public static void inversSPL(Matrix MatrixSPL)	Prosedur ini digunakan untuk mencari solusi SPL dengan kaidah cramer.

3.2.2 Determinan.java

Class ini digunakan untuk mencari determinan dari suatu matriks dengan dua metode, yaitu metode kofaktor dan OBE.

- **Atribut**

-

- **Konstruktor**

-

- **Metode**

Nama Procedure/Function	Fungsi
public static double detKofaktor(Matrix m)	Fungsi ini digunakan untuk mencari determinan suatu matriks dengan menggunakan metode kofaktor. Baris/kolom yang dijadikan acuan dalam metode ini adalah baris 0 / baris pertama.

public static double detOBE(Matrix m)	Fungsi ini digunakan untuk mencari determinan suatu matriks dengan menggunakan metode OBE (operasi baris elementer) / reduksi baris. Matriks dibuat menjadi matriks segitiga bawah dan besar determinan diambil dari perkalian elemen diagonalnya.
--	--

3.2.3 Invers.java

Class ini digunakan untuk mencari matriks invers atau balikan dari suatu matriks dengan dua metode, yaitu metode adjoint dan metode OBE.

- **Atribut**

-

- **Konstruktor**

public double matrix[][]	Elemen-elemen dari matriks dalam bentuk double
--------------------------	--

- **Metode**

Nama Procedure/Function	Fungsi
public static Matrix inversIdentitas (Matrix matrix)	Fungsi yang digunakan untuk mencari invers dari suatu matriks dengan menggunakan metode reduksi baris. Fungsi mengembalikan matriks balikan.
public static Matrix inversAdjoin (Matrix m)	Fungsi yang digunakan untuk mencari invers dari suatu matriks dengan menggunakan rumus $\frac{1}{\det} \times [adjoin]$. Fungsi mengembalikan matriks balikan.

3.2.4 Interpolasi.java

Class ini digunakan untuk mencari hasil interpolasi polinom nilai x tertentu dan fungsinya.

- **Atribut**

-

- **Konstruktor**

-

- **Metode**

Nama Procedure/Function	Fungsi
public static void interpolasiPolinomialKeyboard (Matrix m)	Prosedur ini digunakan untuk mencari fungsi interpolasi polinom dan hasil taksiran x. Inputan titik-titiknya berasal dari ketikan keyboard.
public static void interpolasiPolinomialFile (Matrix m)	Prosedur ini digunakan untuk mencari fungsi interpolasi polinom dan hasil taksiran x. Inputan titik-titiknya berasal dari file.

3.2.5 Bicubic.java

Class ini digunakan untuk mencari hasil interpolasi *bicubic spline* dari suatu matriks.

- **Atribut**

-

- **Konstruktor**

-

- **Metode**

public static Matrix XForF()	Fungsi digunakan untuk mencari nilai matriks X (koefisien a) pada interpolasi <i>bicubic spline</i> . Fungsi ini mencari 4 baris pertama (baris 1 - 4) yaitu koefisien untuk F. Fungsi ini mengembalikan matriks berukuran 4x16.
public static Matrix XForFx()	Fungsi digunakan untuk mencari nilai matriks X (koefisien a) pada interpolasi <i>bicubic spline</i> . Fungsi ini mencari 4 baris selanjutnya (baris 5 - 8) yaitu koefisien untuk Fx (turunan F terhadap x). Fungsi ini mengembalikan matriks 4x16.
public static Matrix XForFy()	Fungsi digunakan untuk mencari nilai matriks X (koefisien a) pada interpolasi <i>bicubic spline</i> . Fungsi ini mencari 4 baris selanjutnya (baris 9 - 12) yaitu koefisien untuk Fy (turunan F terhadap y). Fungsi ini mengembalikan matriks 4x16.
public static Matrix XForFxy()	Fungsi digunakan untuk mencari nilai matriks X (koefisien a) pada interpolasi

	<i>bicubic spline</i> . Fungsi ini mencari 4 baris terakhir (baris 13 - 16) yaitu koefisien untuk F_{xy} (turunan F terhadap x dan y). Fungsi ini mengembalikan matriks 4×16 .
public static Matrix matrixX()	Fungsi digunakan untuk mencari nilai matriks X (koefisien a) pada interpolasi <i>bicubic spline</i> . Fungsi ini menggabungkan semua komponen matriks yang diperoleh melalui fungsi $XForF$, $XForFx$, $XForFy$, dan $XForFxy$, sehingga membentuk matriks X dengan ukuran 16×16 .
public static void interpolasiBicubic(Matrix m)	Prosedur mencari hasil interpolasi bikubik dengan menggunakan rumus $y = Xa$ untuk mencari matriks a , kemudian menggunakan nilai tersebut untuk mencari hasilnya dengan mengalikan elemen matriks a dengan a^i dan b^j . Hasil tersebut kemudian ditampilkan ke layar.

3.2.6 Regresi.java

Class ini digunakan untuk mencari penyelesaian regresi linier berganda.

- **Atribut**

-

- **Konstruktor**

-

- **Metode**

public static void regresiLinearKeyboard(Matrix m)	Prosedur ini mencari hasil taksiran berdasarkan regresi. Prosedur ini menerima inputan matriks dan nilai x yang ingin ditaksir dan data dari keyboard.
public static void regresiLinearFile(Matrix m)	Prosedur ini mencari hasil taksiran berdasarkan regresi. Prosedur ini menerima inputan matriks dan nilai x yang ingin ditaksir dan data dari file.

BAB IV

EKSPERIMEN

4.1 SPL

4.1.1 Test Case 1

Temukan solusi SPL $Ax = b$, berikut:

1) Test Case 1a

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

- Metode Gauss

Solusi tidak ada.

- Metode Gauss-Jordan

Solusi tidak ada.

- Metode Matriks Balikan

SPL tidak dapat diselesaikan dengan metode invers.

- Kaidah Cramer

SPL tidak dapat diselesaikan dengan metode cramer.

2) Test Case 1b

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

- Metode Gauss

```
Solusi banyak (parametrik):
X2 = r
X5 = s
X1 = 3.0000 + r - s
X4 = t
X2 = 1.5000 + 1.5000t + 0.5000s
X4 = u
X4 = -1.0000 + s
```

Pada solusi diatas semua solusi yang parametrik dideklarasi terlebih dahulu baru kemudian digunakan ke solusi lain yang berhubungan. Maka solusi diatas dapat disederhanakan solusinya menjadi

X1 = 3.0000

X2 = 2.0000s

X3 tidak dipedulikan atau dibebaskan karena pada persamaan-persamaannya

X4 = -1.000 + s

X5 = r

- Metode Gauss-Jordan

```
Solusi banyak (parametrik):
X5 = r
X1 = 3.0000 + r
X2 = 2.0000r
X4 = s
X4 = -1.0000 + r
```

Solusi diatas terdapat double output untuk X4. Hal ini dikarenakan X4 dideklarasikan terlebih dahulu sebagai solusi yang parametrik.

- Metode Matriks Balikan

```
SPL tidak dapat diselesaikan dengan metode invers.
```

- Kaidah Cramer

```
SPL tidak dapat diselesaikan dengan metode cramer.
```

3) Test Case 1c

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

- Metode Gauss

```
Solusi banyak (parametrik):
X0 = r
X5 = s
X2 = 2.0000 - s
X6 = t
X2 = 1.0000 - t
X4 = u
X4 = -1.0000 - s
```

Pada solusi diatas terdapat double output pada X4. Hal ini dikarenakan X4 dideklarasikan terlebih dahulu sebagai solusi yang parametrik.

- Metode Gauss-Jordan

```
Solusi banyak (parametrik):
X0 = r
X1 = s
X2 = t
X3 = u
X6 = v
X5 = 1.0000 + v
X2 = 1.0000 - v
X4 = -1.0000 -
```

Pada solusi diatas terdapat double output pada X2. Hal ini dikarenakan X2 dideklarasikan terlebih dahulu sebagai solusi yang parametrik.

- Metode Matriks Balikan

```
SPL tidak dapat diselesaikan dengan metode invers.
```

- Kaidah Cramer

```
SPL tidak dapat diselesaikan dengan metode cramer.
```

4) Test Case 1d

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

H adalah matriks *Hilbert*. Cobakan untuk $n = 6$ dan $n = 10$.

Kasus $n = 6$

- Metode Gauss

```
Solusi tunggal:
X[1] = 35.6558
X[2] = -619.2082
X[3] = 3282.7840
X[4] = -7351.4848
X[5] = 7323.5567
X[6] = -2676.9191
```

- Metode Gauss-Jordan

```
Solusi tunggal:
X[1] = 35.6558
X[2] = -619.2082
X[3] = 3282.7840
X[4] = -7351.4848
X[5] = 7323.5567
X[6] = -2676.9191
```

- Metode Matriks Balikan

```
X1 = 35.6558
X2 = -619.2082
X3 = 3282.7840
X4 = -7351.4848
X5 = 7323.5567
X6 = -2676.9191
```

- Kaidah Cramer

```
X1 = 35.6558
X2 = -619.2075
X3 = 3282.7802
X4 = -7351.4763
X5 = 7323.5482
X6 = -2676.9159
```

Kasus n = 10

- Metode Gauss

```
Solusi tunggal:
X[1] = 12.0218
X[2] = -77.1603
X[3] = 124.5157
X[4] = 57.3643
X[5] = -268.2617
X[6] = 34.5441
X[7] = 226.4496
X[8] = -74.6534
X[9] = 35.0669
X[10] = -72.9123
```

- Metode Gauss-Jordan

```
Solusi tunggal:
X[1] = 12.0218
X[2] = -77.1603
X[3] = 124.5157
X[4] = 57.3643
X[5] = -268.2617
X[6] = 34.5441
X[7] = 226.4496
X[8] = -74.6534
X[9] = 35.0669
X[10] = -72.9123
```

- Metode Matriks Balikan

```
X1 = 12.0218
X2 = -77.1603
X3 = 124.5157
X4 = 57.3643
X5 = -268.2617
X6 = 34.5441
X7 = 226.4496
X8 = -74.6534
X9 = 35.0669
X10 = -72.9123
```

- Kaidah Cramer

```
X1 = 12.0218
X2 = -77.1603
X3 = 124.5157
X4 = 57.3643
X5 = -268.2617
X6 = 34.5441
X7 = 226.4496
X8 = -74.6534
X9 = 35.0669
X10 = -72.9123
```

4.1.2 Test Case 2

SPL berbentuk matriks augmented

- 1) Test Case 2a

$$\left[\begin{array}{ccccc} 1 & -1 & 2 & -1 & -1 \\ 2 & 1 & -2 & -2 & -2 \\ -1 & 2 & -4 & 1 & 1 \\ 3 & 0 & 0 & -3 & -3 \end{array} \right]$$

- Metode Gauss

```
Solusi banyak (parametrik):
X2 = r
X3 = s
X4 = t
X1 = -1.0000 + r - 2.0000s + t
X2 = 2.0000s
```

Pada solusi diatas terdapat double output pada X2. Hal ini dikarenakan X2 dideklarasikan terlebih dahulu sebagai solusi yang parametrik. Solusi diatas dapat disederhanakan menjadi

$$X1 = -1.0000 + t$$

$$X2 = 2.0000s$$

$$X3 = s$$

$$X4 = t$$

- Metode Gauss-Jordan

```
Solusi banyak (parametrik):
X4 = r
X1 = -1.0000 + r
X3 = s
X2 = 2.0000s
```

- Metode Matriks Balikan

```
SPL tidak dapat diselesaikan dengan metode invers.
```

- Kaidah Cramer

SPL tidak dapat diselesaikan dengan metode cramer.

2) Test Case 2b

$$\left[\begin{array}{ccccc} 2 & 0 & 8 & 0 & 8 \\ 0 & 1 & 0 & 4 & 6 \\ -4 & 0 & 6 & 0 & 6 \\ 0 & -2 & 0 & 3 & -1 \\ 2 & 0 & -4 & 0 & -4 \\ 0 & 1 & 0 & -2 & 0 \end{array} \right]$$

- Metode Gauss

```
X3 = r
X1 = 4.0000 - 4.0000r
X4 = s
X2 = 6.0000 - 4.0000s
X3 = 1.0000
X4 = 1.0000
```

Hasil diatas dapat disederhanakan menjadi $x_1 = 0$, $x_2 = 2$, $x_3 = 1$, $x_4 = 1$. Ini diperoleh dengan melakukan substitusi nilai yang terdefinisi ke x yang menggunakan variabel dari nilai tersebut. Contohnya x_3 didefinisikan sebagai r untuk digunakan di x_1 namun x_3 juga bernilai 1 sehingga ini bisa disubstitusi nilainya.

- Metode Gauss-Jordan

```
Solusi tunggal:
X[1] = 0.0000
X[2] = 2.0000
X[3] = 1.0000
X[4] = 1.0000
X[5] = 0.0000
X[6] = 0.0000
```

- Metode Matriks Balikan

SPL tidak dapat diselesaikan dengan metode invers.

- Kaidah Cramer

SPL tidak dapat diselesaikan dengan metode cramer.

4.1.3 Test Case 3

SPL berbentuk

1) Test Case 3a

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

- Metode Gauss

```
Solusi tunggal:
X[1] = -0.2243
X[2] = 0.1824
X[3] = 0.7095
X[4] = -0.2581
```

- Metode Gauss-Jordan

```
Solusi tunggal:
X[1] = -0.2243
X[2] = 0.1824
X[3] = 0.7095
X[4] = -0.2581
```

- Metode Matriks Balikan

```
X1 = -0.2243
X2 = 0.1824
X3 = 0.7095
X4 = -0.2581
```

- Kaidah Cramer

```
X1 = -0.2243
X2 = 0.1824
X3 = 0.7095
X4 = -0.2581
```

2) Test Case 3b

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

- Metode Gauss

```

X2 = r
X3 = s
X1 = 8.0000 - r - s
X4 = t
X5 = u
X6 = v
X7 = w
X8 = x
X2 = 57.2400 - 3.6568s - t - 3.6568u - v - 3.6568w - x
X9 = y
X3 = 344.8356 - u - 17.4866v - w - 17.4866x - 14.3148y
X4 = 15.0000 - u - v
X5 = -3155896505272857100.0000 + 161546268097193376.0000v + 5530060054440778.0000w + 167076328151634144.0000x + 137773989069044560.0000
y
X6 = 19.5356 - 0.0342w - 1.0342x - 0.8528y
X7 = 13.0000 - x - y
X8 = 13.0000 - 17.0000y
X9 = -16.0000

```

Persamaan parametrik tidak akan dapat disederhanakan karena akan menyebabkan setiap solusi X jika disubstitusi-substitusikan akan berdiri sendiri dan tidak bergantung pada variabel lain. Hal ini tentu saja bukan termasuk ke dalam parametrik melainkan solusi tunggal. Namun, saat disubstitusikan masing-masing X nya ke dalam persamaan-persamaan yang ada tidak menghasilkan hasil yang sama dengan persamaan. Jadi, dapat disimpulkan sebenarnya solusi ini juga tidak ada.

- Metode Gauss-Jordan

Solusi tidak ada.

- Metode Matriks Balikan

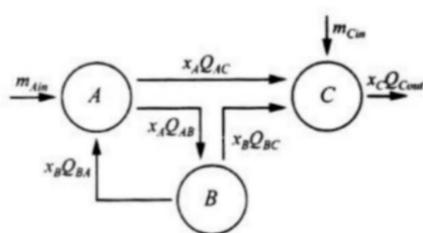
SPL tidak dapat diselesaikan dengan metode invers.

- Kaidah Cramer

SPL tidak dapat diselesaikan dengan metode cramer.

4.1.4 Test Case 4

Lihatlah sistem reaktor pada gambar berikut



Dengan laju volume Q dalam m^3/s dan input massa min dalam mg/s . Konservasi massa pada tiap inti reaktor adalah sebagai berikut:

$$A: m_{A_{in}} + Q_{BA}x_B - Q_{AB}x_A - Q_{AC}x_A = 0$$

$$B: Q_{AB}x_A - Q_{BA}x_B - Q_{BC}x_B = 0$$

$$C: m_{C_{in}} + Q_{AC}x_A + Q_{BC}x_B - Q_{C_{out}}x_C = 0$$

Tentukan solusi x_A , x_B , x_C dengan menggunakan parameter berikut : $Q_{AB} = 40$, $Q_{AC} = 80$, $Q_{BA} = 60$, $Q_{BC} = 20$ dan $Q_{C_{out}} = 150 \text{ } m^3/\text{s}$ dan $m_{A_{in}} = 1300$ dan $m_{C_{in}} = 200 \text{ mg/s}$.

1) Test Case 4

- Metode Gauss

```
Solusi tunggal:  
X[1] = 14.4444  
X[2] = 7.2222  
X[3] = 10.0000
```

- Metode Gauss-Jordan

```
Solusi tunggal:  
X[1] = 14.4444  
X[2] = 7.2222  
X[3] = 10.0000
```

- Metode Matriks Balikan

```
X1 = 14.4444  
X2 = 7.2222  
X3 = 10.0000
```

- Kaidah Cramer

```
X1 = 14.4444  
X2 = 7.2222  
X3 = 10.0000
```

4.2 Interpolasi Polinom

4.2.1 Test Case 5

1) Test Case 5a

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	0.1	0.3	0.5	0.7	0.9	1.1	1.3
f(x)	0.003	0.067	0.148	0.248	0.370	0.518	0.697

Lakukan pengujian pada nilai-nilai berikut:

$$\begin{array}{ll} x = 0.2 & f(x) = ? \\ x = 0.55 & f(x) = ? \\ x = 0.85 & f(x) = ? \\ x = 1.28 & f(x) = ? \end{array}$$

Hasil Test Case 5a :

$$f(x) = -0.0000x^6 + 0.0000x^5 + 0.0260x^4 + 0.0000x^3 + 0.1974x^2 + 0.2400x - 0.0230, f(0.8500) = 0.3372$$

2) Test Case 5b

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

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

Gunakanlah data di atas dengan memanfaatkan interpolasi polinomial untuk melakukan prediksi jumlah kasus baru Covid-19 pada 16/07/2022.

Hasil Test Case 5b :

$$f(x) = -140993.7122x^9 + 9372849.2391x^8 - 275474539.4207x^7 + 4695806315.4288x^6 - 51131876760.1326x^5 + 368550807175.5316x^4 - 1756810186361.3738x^3 + 5334203055240.2830x^2 - 9346993079172.9630x + 7187066071658.6370, f(7.5161) = 53533.6055$$

3) Test Case 5c

Sederhanakan fungsi $f(x)$ yang memenuhi kondisi

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

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 $h = (2 - 0)/5 = 0.4$.

Hasil Test Case 5c :

$$\begin{aligned} f(x) &= 0.2363x^5 - 1.4213x^4 + 3.2371x^3 - 3.5527 \\ &x^2 + 2.0353x - 0.0000, \quad f(4.0000) = 36.5558 \end{aligned}$$

4.3 Regresi Linier Berganda

4.3.1 Test Case 6

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.

- 1) Test Case 6

$$f(x) = -3.5078 - 0.0026x_1 + 0.0008x_2 + 0.1542x_3, \quad f(x_k) = 0.9384$$

4.4 Interpolasi Bicubic Spline

4.4.1 Test Case 7

$$\begin{pmatrix} 21 & 98 & 125 & 153 \\ 51 & 101 & 161 & 59 \\ 0 & 42 & 72 & 210 \\ 16 & 12 & 81 & 96 \end{pmatrix}$$

Tentukan nilai:

$$\begin{aligned} f(0, 0) &= ? \\ f(0.5, 0.5) &= ? \\ f(0.25, 0.75) &= ? \\ f(0.1, 0.9) &= ? \end{aligned}$$

1) Test Case 7a

$$f(0.0, 0.0) = 21.0000$$

2) Test Case 7b

$$f(0.5, 0.5) = 87.7969$$

3) Test Case 7c

$$f(0.25, 0.75) = 82.1482$$

4) Test Case 7d

$$f(0.5, 0.5) = 2.1250$$

4.5 Determinan

4.5.1 Test Case 8

Hitunglah determinan dari matriks-matriks berikut.

1) Test Case 8a

$$21 \ 98 \ 125 \ 153$$

$$51 \ 101 \ 161 \ 59$$

$$0 \ 42 \ 72 \ 210$$

$$16 \ 12 \ 81 \ 96$$

- Metode Kofaktor

$$6781746.0000$$

- Metode OBE

$$6781746.0000$$

2) Test Case 8b

1 3 5 9

1 3 1 7

4 3 9 7

5 2 0 9

- Metode Kofaktor

-376.0000

- Metode OBE

-376.0000**4.6 Invers****4.6.1 Test Case 9**

Carilah matriks invers dari matriks berikut

1) Test Case 9a

1 8 4 3

5 7 1 9

6 6 0 5

2 7 3 1

- Metode OBE

-7.1667	4.1667	-4.8333	8.1667
7.5833	-4.5833	5.4167	-8.5833
-12.7500	7.7500	-9.2500	14.7500
-0.5000	0.5000	-0.5000	0.5000

- Metode Adjoin

-7.1667	4.1667	-4.8333	8.1667
7.5833	-4.5833	5.4167	-8.5833
-12.7500	7.7500	-9.2500	14.7500
-0.5000	0.5000	-0.5000	0.5000

2) Test Case 9b

1 2 3

4 5 6

7 8 9

- Metode OBE

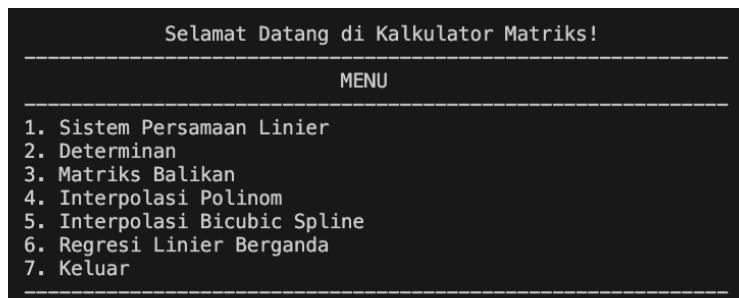
Invers tidak ada.

- Metode Adjoint

Invers tidak ada.

4.7 Main

4.7.1 Menu Pilihan



4.7.2 Menu Metode

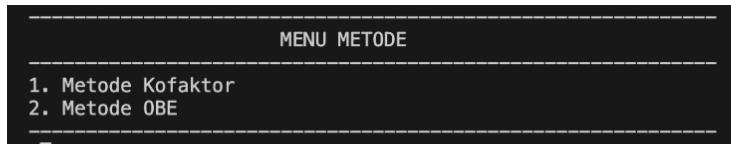
- 1) Invers



- 2) SPL



- 3) Determinan



4.7.3 Menu Output File



4.7.4 Tampilan Keluar

Terima Kasih!
OPET BROT BROT
13522063 | 13522070 | 13522113

BAB V

KESIMPULAN

5.1 Kesimpulan

Dalam menyelesaikan sistem persamaan linier, terdapat empat metode yang dapat kita digunakan, yaitu metode Gauss, metode Gauss-Jordan, metode matriks balikan, dan kaidah cramer. Dalam mencari nilai determinan dari suatu matriks dapat digunakan metode kofaktor dan metode OBE. Dalam mencari matriks balikan, dapat digunakan metode OBE identitas dan metode matriks adjoint.

Metode-metode tersebut dapat digunakan untuk mencari penyelesaian interpolasi polinom, regresi linier berganda, dan interpolasi bicubic spline. Sebagai contoh, metode gauss/gauss-jordan dapat digunakan dalam penyelesaian interpolasi polinom dan regresi linier berganda. Invers dan determinan dapat digunakan dalam penyelesaian interpolasi *bicubic spline*.

Pada tugas besar ini, kita perlu membuat library yang berisikan fungsi-fungsi SPL, determinan, dan invers. Kemudian, kita perlu menggunakan fungsi-fungsi tersebut dalam penyelesaian interpolasi polinom, regresi linier berganda, dan interpolasi *bicubic spline*. Fungsi-fungsi dan implementasinya dibuat dalam bahasa Java.

5.2 Saran dan Refleksi

Dalam penggeraan, diperlukan waktu yang lebih lama dalam beradaptasi dengan bahasa Java, mengingat bahwa bahasa ini cukup baru dan tidak diajarkan secara langsung melalui mata kuliah serta banyak mata kuliah lain dengan penggunaan bahasa yang berbeda dari bahasa Java. Diperlukan juga waktu lebih karena masih baru beradaptasi di dalam lingkup jurusan yang diberikan tugas, kuis, dan lain-lain dalam jumlah banyak dalam waktu bersamaan.

Selain itu, dalam penggeraan tugas besar ini, diperlukan ketelitian dalam pembacaan spek sehingga tidak salah dalam implementasinya. Selain itu, perlu dieksplor lebih banyak skema atau kondisi, sehingga semua kondisi dapat ditemukan penyelesaian yang tepat (*error handling*). Melalui tugas besar ini, kami memahami pentingnya kemampuan kerja sama, koordinasi, dan komunikasi yang baik.

DAFTAR PUSTAKA

Informatika.stei.itb.ac.id. (2023). Sistem persamaan linier (Bagian 1: Metode eliminasi Gauss). Diakses pada 24 September 2022, dari <https://informatika.stei.itb.ac.id/~rinaldi.munir/AljabarGeometri/2023-2024/Algeo-03-Sistem-Persamaan-Linier-2023.pdf>

Informatika.stei.itb.ac.id. (2023). Sistem persamaan linier (Bagian 2: Metode eliminasi Gauss-Jordan). Diakses pada 24 September 2022, dari <https://informatika.stei.itb.ac.id/~rinaldi.munir/AljabarGeometri/2023-2024/Algeo-05-Sistem-Persamaan-Linier-2023.pdf>

Informatika.stei.itb.ac.id. (2023). Sistem persamaan linier (Bagian 2: Metode eliminasi Gauss-Jordan). Diakses pada 24 September 2022, dari <https://informatika.stei.itb.ac.id/~rinaldi.munir/AljabarGeometri/2023-2024/Algeo-04-Tiga-Kemungkinan-Solusi-SPL-2023.pdf>

Informatika.stei.itb.ac.id. (2023). Determinan (Bagian 1). Diakses pada 26 September 2022, dari <https://informatika.stei.itb.ac.id/~rinaldi.munir/AljabarGeometri/2023-2024/Algeo-08-Determinan-bagian1-2023.pdf>

Informatika.stei.itb.ac.id. (2023). Determinan (Bagian 2). Diakses pada 26 September 2022, dari <https://informatika.stei.itb.ac.id/~rinaldi.munir/AljabarGeometri/2023-2024/Algeo-09-Determinan-bagian2-2023.pdf>

Link repository: <https://github.com/BocilBlunder/Algeo01-22063.git>

