

**LAPORAN TUGAS BESAR 01**  
**IF2123 ALJABAR LINIER DAN GEOMETRI**

Kelompok Teman Tuan Mike



Disusun Oleh:

Albertus Christian Poandy	13523077
Grace Evelyn Simon	13523087
Karol Yangqian Poetracahaya	13523093

**PROGRAM STUDI TEKNIK INFORMATIKA**  
**SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA**  
**INSTITUT TEKNOLOGI BANDUNG**  
**2024**

## DAFTAR ISI

<b>BAB 1</b>	<b>3</b>
1.1 Tujuan	3
1.2 Spesifikasi	4
<b>BAB 2</b>	<b>7</b>
2.1 Metode Operasi Baris Elementer	7
2.2 Metode Eliminasi Gauss	7
2.3 Metode Eliminasi Gauss-Jordan	8
2.4 Determinan	9
2.5 Matriks Balikan	10
2.6 Matriks Kofaktor	11
2.7 Matriks Adjoin	11
2.8 Kaidah Cramer	12
2.9 Interpolasi Polinom	13
2.10 Interpolasi Bicubic Spline	14
2.11 Regresi Linier Berganda dan Kuadratik Berganda	16
<b>BAB 3</b>	<b>18</b>
3.1 Matriks.java	18
3.2 Linalg.java	20
3.3 SistemPersamaanLinier.java	22
3.4 Polinomial.java	23
3.5 LinearRegression.java	24
3.6 BicubicSplineInterpolation.java	24
3.7 App.java	26
3.8 Bicubic Spline Controller	27
3.9 DeterminanController.java	29
3.10 FileHandler.java	30
3.11 InterpolasiPolinomialController.java	31
3.12 MainMenuController.java	33
3.13 MatriksBalikanController.java	34
3.14 RegresiBergandaController.java	36
3.15 SPLController.java	39
3.16 TextFlowHandler.java	41
3.17 ImageResizerController.java	42
<b>BAB 4</b>	<b>45</b>
4.1. Sistem Persamaan Linier	45
4.1.1. Kasus 1a	45
4.1.2. Kasus 1b	47
4.1.3. Kasus 1c	48

4.1.4. Kasus 1d	51
4.1.5. Kasus 2a	55
4.1.6. Kasus 2b	58
4.1.7. Kasus 3a	60
4.1.8. Kasus 3b	62
4.1.9. Kasus 4	64
4.2. Interpolasi polinomial	67
4.2.1. Kasus 5a	67
4.2.2. Kasus 5b	69
4.2.3. Kasus 5c	73
Sederhanakan fungsi $f(x)$ yang memenuhi kondisi	73
4.3. Regresi Berganda	73
4.3.1. Kasus 6a	75
4.3.2. Kasus 6b	76
4.4. Interpolasi Bicubic Spline	76
4.4.1. Kasus 7a	77
<b>BAB 5</b>	<b>79</b>
<b>DAFTAR PUSTAKA</b>	<b>82</b>

## **BAB 1**

### **DESKRIPSI MASALAH**

Sistem persamaan linier (SPL) banyak ditemukan di dalam bidang sains dan rekayasa. Dalam penyelesaian sembarang SPL, terdapat berbagai metode yang dapat digunakan, 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). Dalam penyelesaian SPL menggunakan berbagai metode tersebut, digunakan matriks sebagai kunci utama penyelesaiannya. Di dalam Tugas Besar 1 ini, kami diminta membuat satu atau lebih *library* aljabar linier dalam Bahasa Java. Library tersebut 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). Kami juga diminta untuk membuat *library* yang mampu menghitung determinan matriks dengan dua metode, yaitu reduksi baris dan dengan ekspansi kofaktor. Selanjutnya, *library* tersebut digunakan di dalam program Java untuk menyelesaikan berbagai persoalan yang dimodelkan dalam bentuk SPL, menyelesaikan persoalan interpolasi, dan persoalan regresi.

#### **1.1 Tujuan**

Tugas Besar 1 ini dibuat dengan tujuan untuk:

1. Mendapatkan solusi sembarang SPL dengan berbagai 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),
2. Menghitung determinan matriks dengan reduksi baris dan dengan ekspansi kofaktor,
3. Menghitung balikan matriks,
4. Menyelesaikan persoalan interpolasi dan persoalan regresi.

## 1.2 Spesifikasi

1. Program dapat menerima masukan (*input*) baik dari *keyboard* maupun membaca masukan dari *file text*. Untuk SPL, masukan dari *keyboard* adalah m, n, koefisien  $a_{ij}$ , dan  $b_i$ . Masukan dari *file* berbentuk matriks *augmented* tanpa tanda kurung, setiap elemen matriks dipisah oleh spasi. Misalnya,

```
3 4.5 2.8 10 12  
-3 7 8.3 11 -4  
0.5 -10 -9 12 0
```

2. Untuk persoalan menghitung determinan dan matriks balikan, masukan dari *keyboard* adalah n dan koefisien  $a_{ij}$ . Masukan dari *file* berbentuk matriks, setiap elemen matriks dipisah oleh spasi. Misalnya,

```
3 4.5 2.8  
-3 7 8.3  
0.5 -10 -9
```

Luaran (*output*) disesuaikan dengan persoalan (determinan atau invers) dan penghitungan balikan/invers dilakukan dengan metode matriks balikan dan adjoint.

3. Untuk persoalan interpolasi, masukannya jika dari *keyboard* adalah n,  $(x_0, y_0), (x_1, y_1), \dots, (x_n, y_n)$ , dan nilai x yang akan ditaksir nilai fungsinya. Jika masukannya dari *file*, maka titik-titik dinyatakan pada setiap baris tanpa koma dan tanda kurung. Masukan kemudian dilanjutkan dengan satu buah baris berisi satu buah nilai x yang akan ditaksir menggunakan fungsi interpolasi yang telah didefinisikan. Misalnya jika titik-titik datanya adalah (8.0, 2.0794), (9.0, 2.1972), dan (9.5, 2.2513) dan akan mencari nilai y saat x = 8.3, maka di dalam *file text* ditulis sebagai berikut:

```
8.0 2.0794  
9.0 2.1972  
9.5 2.2513  
8.3
```

4. Untuk persoalan regresi, masukannya jika dari *keyboard* adalah n (jumlah peubah x), m (jumlah sampel), semua nilai-nilai  $x_{1i}$ ,  $x_{2i}$ , ...,  $x_{ni}$ , nilai  $y_i$ , dan nilai-nilai  $x_k$  yang akan ditaksir nilai fungsinya. Jika masukannya dari *file*, maka titik-titik dinyatakan pada setiap baris tanpa koma dan tanda kurung.
5. Untuk persoalan SPL, luaran program adalah solusi SPL. Jika solusinya tunggal, tuliskan nilainya. Jika solusinya tidak ada, tuliskan solusi tidak ada, jika solusinya banyak, maka tuliskan solusinya dalam bentuk parametrik (misalnya  $x_4 = -2$ ,  $x_3 = 2s - t$ ,  $x_2 = s$ , dan  $x_1 = t$ ).
6. Untuk persoalan polinom interpolasi dan regresi, luarannya adalah persamaan polinom/regresi dan taksiran nilai fungsi pada x yang diberikan. Contoh luaran untuk interpolasi adalah

$$f(x) = -0.0064x^2 + 0.2266 + 0.6762, \quad f(5) = \dots$$

dan untuk regresi adalah

$$f(x) = -9.5872 + 1.0732x_1, \quad f(x_k) = \dots$$

untuk kasus regresi kuadratik, variabel boleh menggunakan  $x_1$ ,  $x_2$ , dan lain-lain tetapi perlu dijelaskan variabel tersebut merepresentasikan apa. Contoh

$$X_1 = X$$

$$X_3 = X^2$$

$$X_5 = XY$$

[Persamaan dan Solusi]

7. Untuk persoalan *bicubic spline interpolation*, masukan dari *file text (.txt)* yang berisi matriks berukuran  $4 \times 4$  yang berisi konfigurasi nilai fungsi dan turunan berarah disekitaranya, diikuti dengan nilai a dan b untuk mencari nilai  $f(a, b)$ . Misalnya jika nilai dari  $f(0, 0)$ ,  $f(1, 0)$ ,  $f(0, 1)$ ,  $f(1, 1)$ ,  $f_x(0, 0)$ ,  $f_x(1, 0)$ ,  $f_x(0, 1)$ ,  $f_x(1, 1)$ ,  $f_y(0, 0)$ ,  $f_y(1, 0)$ ,  $f_y(0, 1)$ ,  $f_y(1, 1)$ ,  $f_{xy}(0, 0)$ ,  $f_{xy}(1, 0)$ ,  $f_{xy}(0, 1)$ ,  $f_{xy}(1, 1)$  berturut-turut adalah 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16 serta nilai a dan b yang dicari berturut-turut adalah 0.5 dan 0.5 maka isi *file text* ditulis sebagai berikut:

```
1 2 3 4  
5 6 7 8  
9 10 11 12  
13 14 15 16  
0.5 0.5
```

Luaran yang dihasilkan adalah nilai dari  $f(0.5, 0.5)$ .

8. Luaran program harus dapat ditampilkan pada layar komputer dan dapat disimpan ke dalam *file*.
9. Bahasa program yang digunakan adalah Java. Anda bebas untuk menggunakan versi java apapun dengan catatan di atas java versi 8 (8/9/11/15/17/19/20).
10. Program dapat dibuat dengan pilihan menu. Urutan menu dan isinya dipersilakan dirancang masing-masing. Misalnya, menu:

```
MENU  
1. Sistem Persamaan Linier  
2. Determinan  
3. Matriks balikan  
4. Interpolasi Polinom  
5. Interpolasi Bicubic Spline  
6. Regresi linier dan kuadratik berganda  
7. Interpolasi Gambar (Bonus)  
8. Keluar
```

Untuk pilihan menu nomor 1 ada sub-menu lagi yaitu pilihan metode:

```
1. Metode eliminasi Gauss  
2. Metode eliminasi Gauss-Jordan  
3. Metode matriks balikan  
4. Kaidah Cramer
```

Begitu juga untuk pilihan menu nomor 2, 3, dan 6

## **BAB 2**

### **TEORI SINGKAT**

#### **2.1 Metode Operasi Baris Elementer**

Operasi Baris Elementer (OBE) merupakan suatu operasi yang diterapkan pada baris suatu matriks. OBE dapat digunakan untuk menentukan invers suatu matriks dan menyelesaikan suatu sistem persamaan linier (SPL). OBE diterapkan terhadap matriks *augmented* (matriks yang diperbesar dengan cara menambahkan kolom tambahan yang entrinya adalah bilangan ruas kanan dari persamaan SPL) sehingga terbentuklah matriks eselon baris (matriks yang memiliki 1 utama (*leading one*) pada setiap baris, kecuali baris yang seluruhnya nol) atau matriks eselon baris tereduksi (matriks eselon baris yang memiliki sifat setiap kolom yang memiliki 1 utama (*leading one*) bernilai nol di tempat lainnya). Tiga OBE yang dapat diterapkan terhadap matriks *augmented* adalah:

1. Kalikan sebuah baris dengan konstanta tidak nol,
2. Pertukarkan dua buah baris,
- 3.Tambahkan sebuah baris dengan kelipatan baris lainnya.

#### **2.2 Metode Eliminasi Gauss**

Metode Eliminasi Gauss merupakan metode yang digunakan untuk menyelesaikan suatu sistem persamaan linier (SPL) dengan cara mengubah SPL menjadi bentuk matriks *augmented* kemudian mengoperasikannya dengan serangkaian OBE untuk menghasilkan matriks eselon baris. Setelah mendapatkan matriks eselon baris, matriks *augmented* tersebut dikembalikan ke bentuk SPL dan dilakukan eliminasi-substitusi untuk mendapat solusi. Meskipun sudah ada sejak lama, metode ini jarang dikenal sebelum kemudian dipopulerkan oleh seorang matematikawan yang berasal dari Jerman, Carl Friedrich Gauss, yang menggunakan metode ini untuk menghitung orbit dari asteroid Ceres dari data yang terbatas. Sifat-sifat matriks eselon baris (matriks dengan 1 utama/*leading one* pada setiap baris, kecuali baris yang seluruhnya nol):

1. Jika sebuah baris tidak terdiri dari seluruhnya nol, maka bilangan tidak nol pertama di dalam baris tersebut adalah 1 (disebut 1 utama),

2. Jika ada baris yang seluruhnya nol, maka semua baris itu dikumpulkan pada bagian bawah matriks,
3. Di dalam dua baris berturutan yang tidak seluruhnya nol, maka 1 utama pada baris yang lebih rendah terdapat lebih jauh ke kanan daripada 1 utama pada baris yang lebih tinggi.

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

Gambar 1. Metode Eliminasi Gauss

### 2.3 Metode Eliminasi Gauss-Jordan

Metode Eliminasi Gauss-Jordan merupakan pengembangan dari Metode Eliminasi Gauss yang dipopulerkan oleh seorang *engineer* Jerman bernama Wilhelm Jordan dalam *handbook* mengenai geodesi berjudul *Handbuch der Vermessungskunde* yang dipublikasikan pada tahun 1888. Metode ini dilakukan dengan mengoperasikan matriks *augmented* sehingga menghasilkan matriks eselon baris tereduksi. Pada metode ini, tidak diperlukan lagi substitusi secara mundur untuk memperoleh nilai-nilai variabel karena nilainya langsung diperoleh dari matriks *augmented* akhir (jika solusinya unik). Metode Eliminasi Gauss-Jordan terdiri dari dua fase:

1. Fase maju (*forward phase*) atau fase Eliminasi Gauss

$$\left[ \begin{array}{cccc} 2 & 3 & -1 & 5 \\ 4 & 4 & -3 & 3 \\ -2 & 3 & -1 & 1 \end{array} \right] \sim_{\text{OBE}} \dots \sim \left[ \begin{array}{cccc} 1 & 3/2 & -1/2 & 5/2 \\ 0 & 1 & 1/2 & 7/2 \\ 0 & 0 & 1 & 3 \end{array} \right]$$

Gambar 2. Fase Maju pada Metode Eliminasi Gauss-Jordan

2. Fase mundur (*backward phase*)

$$\left[ \begin{array}{cccc} 1 & 3/2 & -1/2 & 5/2 \\ 0 & 1 & 1/2 & 7/2 \\ 0 & 0 & 1 & 3 \end{array} \right] \sim \left[ \begin{array}{cccc} 1 & 0 & -5/4 & -11/4 \\ 0 & 1 & 1/2 & 7/2 \\ 0 & 0 & 1 & 3 \end{array} \right] \sim \left[ \begin{array}{cccc} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 2 \\ 0 & 0 & 1 & 3 \end{array} \right]$$

R1 - (3/2)R2      R1 + (5/4)R3      R2 - (1/2)R3

Gambar 3. Fase Mundur pada Metode Eliminasi Gauss-Jordan

## 2.4 Determinan

Dalam bidang aljabar linier, determinan merupakan nilai yang dapat dihitung dari unsur suatu matriks persegi, yaitu matriks dengan jumlah baris dan kolom yang sama. Determinan matriks  $A$  dapat dilambangkan dengan  $\det(A)$ ,  $\det A$ , atau  $|A|$ . Terdapat beberapa metode yang dapat digunakan untuk menghitung determinan suatu matriks:

1. Metode reduksi baris

Determinan suatu matriks dapat diperoleh dengan melakukan OBE pada matriks sampai diperoleh matriks segitiga. Matriks segitiga sendiri dibedakan menjadi dua, yaitu:

- Matriks segitiga atas (*upper triangular*): semua elemen di bawah diagonal utama adalah nol

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ 0 & a_{22} & a_{23} & a_{24} \\ 0 & 0 & a_{33} & a_{34} \\ 0 & 0 & 0 & a_{44} \end{bmatrix} \rightarrow \det(A) = a_{11}a_{22}a_{33}a_{44}$$

Gambar 4. Matriks Segitiga Atas

- Matriks segitiga bawah (*lower triangular*): semua elemen di atas diagonal utama adalah nol

$$A = \begin{bmatrix} a_{11} & 0 & 0 & 0 \\ a_{21} & a_{22} & 0 & 0 \\ a_{31} & a_{32} & a_{33} & 0 \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix} \rightarrow \det(A) = a_{11}a_{22}a_{33}a_{44}$$

Gambar 5. Matriks Segitiga Bawah

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

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

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

$p$  menyatakan banyaknya operasi pertukaran baris di dalam OBE

\*) Asumsi tidak ada operasi perkalian baris dengan konstanta  $k$

- Jika selama reduksi baris ada OBE berupa perkalian baris-baris matriks dengan  $k_1, k_2, \dots, k_m$ , maka

$$\text{maka } \det(A) = \frac{(-1)^p a'_{11} a'_{22} \dots a'_{nn}}{k_1 k_2 \dots k_m}$$

Gambar 6. Menghitung Determinan Matriks

## 2. Metode Ekspansi Kofaktor

Determinan matriks juga dapat diperoleh menggunakan Metode Ekspansi Kofaktor, misal didefinisikan bahwa terdapat matriks berukuran  $n \times n$ :

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

Gambar 7. Matriks Berukuran  $n \times n$

$M_{ij}$  = 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  $a_{ij}$

Determinan matriks  $A$  tersebut dapat dihitung dengan salah satu dari persamaan berikut:

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

Gambar 8. Metode Perhitungan Matriks Kofaktor

## 2.5 Matriks Balikan

Matriks balikan (*invers*) merupakan matriks yang jika dikalikan dengan matriks aslinya akan menghasilkan sebuah matriks identitas (matriks yang komponen diagonal utamanya bernilai satu, sedangkan komponen lainnya bernilai nol). Dapat dituliskan bahwa matriks balikan dari matriks  $A$  merupakan  $(A)^{-1}$  sedemikian sehingga  $A(A)^{-1} = (A)^{-1}A = I$ . Matriks balikan dapat dicari dengan beberapa metode:

### 1. Metode Eliminasi Gauss-Jordan

Dengan cara mengubah matriks  $[A|I]$  sedemikian rupa dengan mengoperasikannya dengan OBE hingga didapatkan  $[I|(A)^{-1}]$ . Apabila ditemukan baris dengan seluruh elemen bernilai nol, dapat disimpulkan bahwa matriks tersebut tidak memiliki balikan.

$$\begin{array}{c} \textcolor{red}{G-J} \\ [A|I] \sim [I|A^{-1}] \end{array}$$

Gambar 9. Matriks Balikan dengan Metode Eliminasi Gauss-Jordan

## 2. Adjoin

Balikan dari suatu matriks  $A$  dapat dihitung dengan rumus:

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

Gambar 10. Matriks Balikan dengan Metode Adjoin

Dengan adjoint  $A$  dapat dicari melalui transpose dari matriks kofaktor  $A$ .

**2.6 Matriks Kofaktor**

Matriks kofaktor merupakan matriks yang memiliki elemen-elemen yang tersusun atas kofaktor entri  $a_{ij}$ , yaitu  $C_{ij}$  atau  $(-1)^{i+j}M_{ij}$  dengan  $M_{ij}$  adalah minor entri  $a_{ij}$  atau determinan submatriks yang elemen-elemennya tidak terletak pada baris  $i$  dan kolom  $j$ .

$$\begin{aligned} A &= \begin{bmatrix} 3 & 2 & -1 \\ 1 & 6 & 3 \\ 2 & -4 & 0 \end{bmatrix} \\ &\left[ + \begin{vmatrix} 6 & 3 \\ -4 & 0 \end{vmatrix} \quad - \begin{vmatrix} 1 & 3 \\ 2 & 0 \end{vmatrix} \quad + \begin{vmatrix} 1 & 6 \\ 2 & -4 \end{vmatrix} \right. \\ &\left. - \begin{vmatrix} 2 & -1 \\ -4 & 0 \end{vmatrix} \quad + \begin{vmatrix} 3 & -1 \\ 2 & 0 \end{vmatrix} \quad - \begin{vmatrix} 3 & 2 \\ 2 & -4 \end{vmatrix} \right] = \begin{bmatrix} 12 & 6 & -16 \\ 4 & 2 & 16 \\ 12 & -10 & 16 \end{bmatrix} \end{aligned}$$

Gambar 11. Contoh Penyelesaian Matriks Kofaktor A

**2.7 Matriks Adjoin**

Matriks adjoint merupakan transpose (pertukaran elemen pada baris menjadi kolom atau kolom menjadi baris) dari suatu matriks dengan elemen-elemennya merupakan kofaktor

dari elemen-elemen matriks A (matriks kofaktor A). Matriks adjoint kemudian dapat digunakan untuk mencari matriks balikan (*invers*) melalui rumus:

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

Gambar 12. Matriks Balikan dengan Metode Adjoin

$$\text{adj}(A) = \begin{bmatrix} C_{11} & C_{12} & C_{13} \\ C_{21} & C_{22} & C_{23} \\ C_{31} & C_{32} & C_{33} \end{bmatrix}^T = \begin{bmatrix} C_{11} & C_{21} & C_{31} \\ C_{12} & C_{22} & C_{32} \\ C_{13} & C_{23} & C_{33} \end{bmatrix}$$

Gambar 13. Adjoin Matriks

## 2.8 Kaidah *Cramer*

Kaidah *cramer* merupakan metode yang dapat digunakan untuk menyelesaikan SPL dengan banyak persamaan dan banyak variabel. Namun, kaidah *Cramer* hanya berlaku ketika SPL memiliki solusi unik/tunggal. Metode kaidah *Cramer* memanfaatkan determinan matriks yang terbentuk dari koefisien (dari SPL) dan determinan matriks lain yang diperoleh dengan mengganti salah satu kolom matriks koefisien dengan vektor yang berada di sebelah kanan persamaan. Dimisalkan  $Ax = b$  adalah SPL yang terdiri dari n persamaan linier dengan n peubah (*variable*) sedemikian sehingga  $\det(A) \neq 0$ , maka SPL tersebut memiliki solusi yang unik, yaitu:

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

Gambar 14. Penyelesaian SPL dengan Kaidah *Cramer*

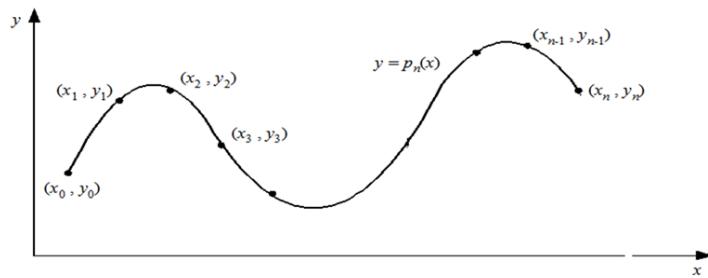
yang dalam hal ini,  $A_j$  adalah matriks yang diperoleh dengan mengganti entri pada kolom ke-j dari A dengan entri dari matriks:

$$\mathbf{b} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix}$$

Gambar 15. Matriks Hasil SPL

## 2.9 Interpolasi Polinom

Interpolasi polinom merupakan teknik interpolasi dengan mengasumsikan pola data yang dimiliki mengikuti pola polinomial baik berderajat satu (linier) maupun berderajat tinggi. Interpolasi polinom digunakan untuk mengaproksimasi atau memprediksi fungsi dan nilainya berdasarkan titik-titik yang diketahui. Dimisalkan terdapat  $n+1$  buah titik berbeda, yaitu dari  $(x_0, y_0), (x_1, y_1), \dots, (x_n, y_n)$ , dapat ditentukan polinom  $p_n(x)$  yang menginterpolasi (melewati) semua titik-titik tersebut sedemikian rupa sehingga  $y_i = p_n(x_i)$  untuk  $i = 0, 1, 2, \dots, n$ .



Gambar 16. Ilustrasi Beberapa Titik yang Diinterpolasi Secara Polinomial

Nilai  $y_i$  dapat berasal dari fungsi matematika  $f(x)$ , seperti  $\ln x$ ,  $\sin x$ , dan sebagainya, sedemikian sehingga  $y_i = f(x_i)$ , sedangkan  $p_n(x)$  disebut fungsi hampiran terhadap  $f(x)$ , atau  $y_i$  berasal dari nilai empiris yang diperoleh melalui percobaan atau pengamatan. Setelah polinom interpolasi  $p_n(x)$  ditemukan,  $p_n(x)$  dapat digunakan untuk menghitung perkiraan nilai  $y$  di sembarang titik di dalam selang  $[x_0, x_n]$ . Polinom interpolasi derajat  $n$  yang menginterpolasi titik-titik  $(x_0, y_0), (x_1, y_1), \dots, (x_n, y_n)$  adalah berbentuk  $p_n(x) = a_0 + a_1x + a_2x^2 + \dots + a_nx^n$ . Jika hanya ada dua titik,  $(x_0, y_0)$  dan  $(x_1, y_1)$ , maka polinom yang menginterpolasi kedua titik tersebut adalah  $p_1(x) = a_0 + a_1x$  yaitu berupa persamaan garis lurus. Jika tersedia tiga titik,  $(x_0, y_0)$ ,  $(x_1, y_1)$ , dan  $(x_2, y_2)$ , maka polinom yang menginterpolasi ketiga titik tersebut adalah  $p_2(x) = a_0 + a_1x + a_2x^2$  atau persamaan kuadrat dan kurvanya berupa parabola. Jika tersedia empat titik,  $(x_0, y_0)$ ,  $(x_1, y_1)$ ,  $(x_2, y_2)$ , dan  $(x_3, y_3)$ , polinom yang menginterpolasi keempat titik tersebut adalah  $p_3(x) = a_0 + a_1x + a_2x^2 + a_3x^3$ , demikian seterusnya. Dengan cara yang sama kita dapat membuat polinom interpolasi berderajat  $n$  untuk  $n$  yang lebih tinggi asalkan tersedia  $(n+1)$  buah titik data. Dengan menyulihkan  $(x_i, y_i)$  ke dalam persamaan polinom  $p_n(x) = a_0 + a_1x + a_2x^2 + \dots + a_nx^n$  untuk  $i = 0, 1, 2, \dots, n$ , akan diperoleh  $n$  buah sistem persamaan lanjar dalam  $a_0, a_1, a_2, \dots, a_n$ ,

$$a_0 + a_1x_0 + a_2x_0^2 + \dots + a_nx_0^n = y_0$$

$$a_0 + a_1 x_1 + a_2 x_1^2 + \dots + a_n x_1^n = y_1$$

...

...

$$a_0 + a_1 x_n + a_2 x_n^2 + \dots + a_n x_n^n = y_n$$

Solusi sistem persamaan lanjar ini, yaitu nilai  $a_0, a_1, \dots, a_n$ , diperoleh dengan menggunakan metode eliminasi Gauss yang sudah dipelajari. Penyelesaian sistem persamaan linier tersebut dapat dilakukan dengan merepresentasikan sistem persamaannya menjadi sebuah matriks. Jika semua konstanta  $a_0, a_1, \dots, a_n$  sudah diketahui nilainya, dapat dibentuk persamaan interpolasi yang dapat memperkirakan nilai  $y$  di sembarang titik dalam selang  $[x_o, x_n]$ .

## 2.10 Interpolasi Bicubic Spline

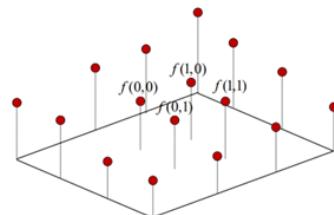
*Bicubic spline interpolation* adalah metode interpolasi yang digunakan untuk mengaproksimasi fungsi di antara titik-titik data yang diketahui. *Bicubic spline interpolation* melibatkan konsep *spline* dan konstruksi serangkaian polinomial kubik di dalam setiap sel segi empat dari data yang diberikan. Pendekatan ini menciptakan permukaan yang halus dan kontinu, memungkinkan untuk perluasan data secara visual yang lebih akurat daripada metode interpolasi linear. Dalam pemrosesan menggunakan interpolasi *bicubic spline* digunakan 16 buah titik, 4 titik referensi utama di bagian pusat, dan 12 titik di sekitarnya sebagai aproksimasi turunan dari keempat titik referensi untuk membangun permukaan bikubik. Bentuk pemodelannya adalah sebagai berikut.

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

$f(0,1), f(1,1)$

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

Solve:  $a_{ij}$



Gambar 17. Pemodelan Interpolasi Bicubic Spline

Selain melibatkan model dasar, juga digunakan model turunan berarah dari kedua sumbu, baik terhadap sumbu  $x$ , sumbu  $y$ , maupun keduanya. Persamaan polinomial yang digunakan adalah sebagai berikut.

$$f(x, y) = \sum_{i=0}^3 \sum_{j=0}^3 a_{ij} x^i y^j$$

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

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

$$f_{xy}(x, y) = \sum_{j=0}^3 \sum_{i=0}^3 a_{ij} i j x^{i-1} y^{j-1}$$

Gambar 18. Persamaan Polinomial Model Turunan Berarah

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

$$\begin{matrix} y = Xa \\ \left[ \begin{array}{c} f(0,0) \\ f(1,0) \\ f(0,1) \\ f(1,1) \\ f_x(0,0) \\ f_x(1,0) \\ f_x(0,1) \\ f_x(1,1) \\ f_y(0,0) \\ f_y(1,0) \\ f_y(0,1) \\ f_y(1,1) \\ f_{xy}(0,0) \\ f_{xy}(1,0) \\ f_{xy}(0,1) \\ f_{xy}(1,1) \end{array} \right] = \left[ \begin{array}{cccccccccccccccc} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 2 & 3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 2 & 3 & 0 & 1 & 2 & 3 & 0 & 1 & 2 & 3 & 0 & 1 & 2 & 3 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 2 & 2 & 2 & 2 & 3 & 3 & 3 & 3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 2 & 3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 2 & 0 & 0 & 0 & 3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 2 & 3 & 0 & 2 & 4 & 6 & 0 & 3 & 6 & 9 \end{array} \right] \begin{array}{l} a_{00} \\ a_{10} \\ a_{20} \\ a_{30} \\ a_{01} \\ a_{11} \\ a_{21} \\ a_{31} \\ a_{02} \\ a_{12} \\ a_{22} \\ a_{32} \\ a_{03} \\ a_{13} \\ a_{23} \\ a_{33} \end{array} \end{matrix}$$

Gambar 19. Persamaan Penyelesaian Matriks Solusi X

Perlu diketahui bahwa elemen pada matriks  $X$  adalah nilai dari setiap komponen koefisien  $a_{ij}$  yang diperoleh dari persamaan fungsi maupun persamaan turunan yang telah dijelaskan sebelumnya. Sebagai contoh, elemen matriks  $X$  pada baris 8 kolom ke 2 adalah koefisien dari  $a_{10}$  pada ekspansi sigma untuk  $f_x(1, 1)$  sehingga diperoleh nilai konstanta  $1 \times 1^{1-1} \times 1^0 = 1$ , sesuai dengan isi matriks  $X$ . Nilai dari vektor  $a$  dapat dicari dari persamaan  $y = Xa$ , lalu vektor  $a$  tersebut digunakan sebagai nilai variabel dalam  $f(x, y)$ , sehingga terbentuk fungsi *interpolasi bicubic* sesuai model. Sama seperti interpolasi polinom, untuk menyelesaikan persoalan interpolasi *bicubic spline* perlu didapat terlebih dahulu setiap komponen koefisien

$a_{ij}$ . Apabila sudah didapat setiap konstanta, dapat diaproksimasi nilai  $z$  di setiap titik  $(x, y)$  dimana  $x$  dan  $y$  dinormalisasi dalam rentang  $[0,1]$ .

## 2.11 Regresi Linier Berganda dan Kuadratik Berganda

Regresi merupakan salah satu metode untuk memprediksi nilai selain menggunakan interpolasi polinom. Beberapa contoh regresi yaitu regresi linier berganda dan regresi kuadratik berganda.

### 1. Regresi Linier Berganda

Regresi linier berganda merupakan perpanjangan dari regresi linier sederhana dimana ada lebih dari satu variabel independen yang digunakan untuk memprediksi variabel dependen. Meskipun sudah ada persamaan jadi untuk menghitung regresi linear sederhana, terdapat persamaan umum dari regresi linear yang bisa digunakan untuk regresi linear berganda, yaitu:

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

Untuk mendapat nilai dari setiap  $\beta_i$  dapat digunakan *Normal Estimation Equation for Multiple Linear Regression* sebagai berikut:

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

Gambar 20. *Normal Estimation Equation for Multiple Linear Regression*

Sistem persamaan linier tersebut kemudian akan diubah ke dalam bentuk matriks SPL dan dapat diselesaikan dengan metode eliminasi Gauss. Setelah didapatkan nilai dari setiap  $\beta_i$  melalui metode *Normal Estimation Equation for Multiple Linear Regression*, dapat dibentuk persamaan regresi linear berganda yang kemudian dapat digunakan untuk mengaproksimasi suatu nilai  $y$  dari suatu nilai  $x_k$ .

### 2. Regresi Kuadratik Berganda

Regresi kuadratik berganda adalah perluasan dari regresi linier berganda, dimana model memasukkan komponen non-linier (terutama kuadrat) dari variabel independen untuk

menangkap hubungan yang dapat bersifat polinomial/melengkung. Ada tiga bentuk persamaan dari regresi kuadratik, yaitu:

- Variabel Linier: variabel dengan derajat satu seperti  $X$ ,  $Y$ , dan  $Z$
- Variabel Kuadrat: variabel dengan derajat dua seperti  $X^2$
- Variabel Interaksi: 2 variabel dengan derajat satu yang dikalikan dengan satu sama lain seperti  $XY$ ,  $YZ$ , dan  $XZ$

Setiap n-peubah, jumlah variabel linier, kuadrat, dan interaksi akan berbeda-beda, contohnya sebagai berikut:

$$\begin{pmatrix} N & \sum u_i & \sum v_i & \sum u_i^2 & \sum u_i v_i & \sum v_i^2 \\ \sum u_i & \sum u_i^2 & \sum u_i v_i & \sum u_i^3 & \sum u_i^2 v_i & \sum u_i v_i^2 \\ \sum v_i & \sum u_i v_i & \sum v_i^2 & \sum u_i^2 v_i & \sum u_i v_i^2 & \sum v_i^3 \\ \sum u_i^2 & \sum u_i^3 & \sum u_i^2 v_i & \sum u_i^4 & \sum u_i^3 v_i & \sum u_i^2 v_i^2 \\ \sum u_i v_i & \sum u_i^2 v_i & \sum u_i v_i^2 & \sum u_i^3 v_i & \sum u_i^2 v_i^2 & \sum u_i v_i^3 \\ \sum v_i^2 & \sum u_i v_i^2 & \sum v_i^3 & \sum u_i^2 v_i^2 & \sum u_i v_i^3 & \sum v_i^4 \end{pmatrix} \begin{pmatrix} a \\ b \\ c \\ d \\ e \\ f \end{pmatrix} = \begin{pmatrix} \sum y_i \\ \sum y_i u_i \\ \sum y_i v_i \\ \sum y_i u_i^2 \\ \sum y_i u_i v_i \\ \sum y_i v_i^2 \end{pmatrix}$$

Gambar 21. Matriks *Quadratic Linear Regression*

$N$  menandakan jumlah peubah, terdapat 2 variabel linier yaitu  $u_i$  dan  $v_i$ , 2 variabel kuadrat yaitu  $u_i^2$  dan  $v_i^2$ , dan 1 variabel interaksi yaitu  $uv$ . Untuk setiap n-peubah, akan terdapat 1 konstan  $N$ ,  $n$  variabel linier,  $n$  variabel kuadrat, dan  $C_2^n$  variabel linier (dengan syarat  $n > 1$ ).

Tentu dengan bertambahnya peubah  $n$ , ukuran matriks akan bertumbuh lebih besar dibandingkan regresi linier berganda tetapi solusi tetap bisa didapat dengan menggunakan SPL. Untuk mendapat nilai konstanta dari setiap  $\beta_i$  pada regresi kuadratik berganda, perlu ditambahkan terlebih dahulu matriks  $X$  sehingga terdapat variabel linier, variabel kuadratik, dan variabel interaksi. Kemudian, setelah didapatkan nilai dari setiap  $\beta_i$  melalui metode *Normal Estimation Equation for Multiple Linear Regression*, dapat dibentuk persamaan regresi kuadratik berganda yang kemudian dapat digunakan untuk mengaproksimasi suatu nilai  $y$  dari suatu nilai  $x_k$ .

**BAB 3****IMPLEMENTASI PUSTAKA DAN PROGRAM DALAM JAVA**

Pustaka yang dibuat berisi ADT (*Abstract Data Type*) Matriks beserta fungsi-fungsi primitifnya serta ADT lainnya yang menggunakan ADT Matriks sebagai pondasinya. Terdapat 6 *class* yang diimplementasikan dalam program ini, yakni Matriks yang berisi seluruh fungsi-fungsi untuk memodifikasi matriks (menghitung baris, menghitung kolom, menghilangkan suatu baris atau kolom, concat antara dua matriks, dan sebagainya), Linalg yang berisi fungsi-fungsi untuk melakukan operasi antara suatu matriks dengan matriks lainnya (pertambahan matriks, pengurangan matriks, perkalian matriks, pembagian matriks, dan sebagainya), SistemPersamaanLinier yang berisi fungsi-fungsi untuk mendapatkan solusi dari sistem persamaan linier melalui 4 buah metode, yakni Eliminasi Gauss, Eliminasi Gauss-Jordan, Balikan, dan Cramer, Interpolasi yang berisi fungsi untuk menghitung interpolasi polinomial, LinearRegression yang berisi fungsi untuk menghitung regresi linier berganda dan regresi kuadratik berganda, dan BicubicSplineInterpolation yang berisi fungsi untuk menghitung interpolasi *bicubic spline*. Tim peneliti juga menciptakan kelas-kelas *controller* untuk mengimplementasikan GUI, serta menciptakan fungsi tambahan berupa *image resizing* dengan mengaplikasikan *bicubic spline interpolation*.

**3.1 Matriks.java**

<b>Attribute</b>
<pre>// Matriks bernama Mat public double[][] Mat // banyak baris dan kolom Matriks private int row, col</pre>
<b>Constructor</b>
<pre>// membuat matriks kosong dengan ukuran 0x0 public Matriks() // membuat matriks dengan ukuran rowxcol</pre>

```
public Matriks(int row, int col)
// membuat matriks baru hasil copy matriks M

public Matriks(Matriks M)
// menerima array dua dimensi bertipe double dan menggunakannya
untuk inisialisasi objek matriks

public Matriks(double[][] arrayOfArrayDouble)
```

### Class Methods

```
// mengembalikan jumlah baris pada matriks
public int getRow()

// mengembalikan jumlah kolom pada matriks
public int getCol()

// menampilkan matriks ke terminal
public void printMatriks()

// mengganti elemen tertentu pada kolom
public Matriks replaceColumn(int col, double[] N)

// mengganti elemen tertentu pada baris
public Matriks replaceRow(int row, double[] N)

// mengisi setiap elemen matriks dengan nilai x
public void fill(double x)

// menghapus baris dan kolom tertentu dari matriks
public Matriks removeRowColMatriks(int row, int col)

// membuat nilai -0.00 menjadi 0.00
public void makePositiveZero()

// menghapus baris terbawah sebanyak numRowRemoved
public Matriks removeRow(int numRowRemoved)

// menghapus kolom terkanan sebanyak numRowRemoved
public Matriks removeCol(int numColRemoved)

// menambahkan baris sebanyak numRowAdded dan baris-baris baru
diisi dengan 0
public Matriks addRowZero(int numRowAdded)

// menambahkan kolom sebanyak numColAdded dan kolom-kolom baru
```

```

diisi dengan 0
public Matriks addColZero(int numColAdded)
// menggabungkan dua matriks
public Matriks concat(Matriks M, boolean isRight)
// membuat matriks identitas
public Matriks identityMatrix()
// menghapus baris tertentu dari matriks
public Matriks popRow(int row)
// menghapus kolom tertentu dari matriks
public Matriks popCol(int col)
// mengambil baris tertentu dari matriks
public double[] getRowElements(int row)
// mengambil kolom tertentu dari matriks
public double[] getColElements(int col)

```

### 3.2 Linalg.java

<b>Attribute</b>
-
<b>Constructor</b>
-
<b>Class Methods</b>
<i>// mengalikan matriks dengan konstanta x</i> <b>public Matriks kaliXMatriks(Matriks Mat, double x)</b> <i>// membagikan matriks dengan konstanta x</i> <b>public Matriks bagiXMatriks(Matriks Mat, double x)</b> <i>// melakukan operasi perkalian matriks (instance * M2)</i> <b>public Matriks perkalianMatriks(Matriks M1, Matriks M2)</b> <i>// melakukan operasi penjumlahan matriks</i> <b>public Matriks penjumlahanMatriks(Matriks M1, Matriks M2)</b>

```
// melakukan operasi transpose matriks
public Matriks transposeMatriks(Matriks Mat)
// membuat matriks kofaktor
public Matriks kofaktorMatriks(Matriks Mat)
// membuat matriks adjoint
public Matriks adjointMatriks(Matriks Mat)
// menghitung determinan matriks
public double determinanMatriks(Matriks M, String method)
// menghitung invers matriks
public Matriks inversMatriks(Matriks Mat, String method)
// mengubah menjadi matriks eselon baris
public Matriks toEselonBaris(Matriks Mat)
// mengubah menjadi matriks eselon baris tereduksi
public Matriks toEselonBarisTereduksi(Matriks Mat)
// mereduksi kolom ke bawah
private Matriks reduksiKolomKeBawah(Matriks Mat, int
leadingOneRow, int leadingOneCol)
// mereduksi kolom ke atas
private Matriks reduksiKolomKeAtas(Matriks Mat, int leadingOneRow,
int leadingOneCol)
// menukar baris pada matriks
public Matriks tukarBaris(Matriks Mat, int row1, int row2)
// mengalikan baris pada matriks dengan skalar x
public Matriks kalikanBaris(Matriks Mat, int row, double x)
// menjumlahkan baris dengan kelipatan baris lainnya
public Matriks jumlahKelipatanBaris(Matriks Mat, int row1, int
row2, double x)
// menghitung invers matriks dengan metode adjoin
private Matriks inversAdjoint(Matriks Mat)
// menghitung invers matriks dengan metode OBE
private Matriks inversOBE(Matriks Mat)
// menghitung determinan matriks dengan metode kofaktor
```

```
private double detKofaktor(Matriks Mat)
// menghitung determinan matriks dengan metode reduksi baris
private double detReduksi(Matriks Mat)
```

### 3.3 SistemPersamaanLinier.java

<b>Attribute</b>
-
<b>Constructor</b>
-
<b>Class Methods</b>
<pre><i>// menyelesaikan SPL dengan Metode Gauss</i> <b>public Matriks metodeGauss(Matriks Mat)</b> <i>// menyelesaikan SPL dengan Metode Gauss-Jordan</i> <b>public Matriks metodeGaussJordan(Matriks Mat)</b> <i>// menyelesaikan SPL dengan Metode Cramer</i> <b>public Matriks metodeCramer(Matriks Mat)</b> <i>// menyelesaikan SPL dengan Metode Invers</i> <b>public Matriks metodeInversMatriks(Matriks Mat)</b> <i>// menyusun matriks sesuai dengan baris Leading one</i> <b>private Matriks sesuaikanBarisLeadingOne(Matriks M)</b> <i>// mengecek apakah ada solusi dari SPL</i> <b>private boolean solutionExist(Matriks A, double[] B)</b> <i>// mencari index pertama yang bernilai tidak nol dari array</i> <b>private int indexNotZero(double[] arr)</b> <i>// menghasilkan matriks solusi dari SPL</i> <b>private Matriks generateSolution(Matriks A, double[] B)</b></pre>

### 3.4 Polinomial.java

<b>Attribute</b>
<pre>// derajat polinomial <b>private int</b> degree  // koefisien persamaan polinomial <b>private double[]</b> coefficients  // mengembalikan derajat polinomial <b>public int</b> getDegree()  // mengambil koefisien polinomial <b>public double[]</b> getCoefficients()</pre>
<b>Constructor</b>
<pre>// membentuk polinomial konstan 0 <b>public Polinomial()</b>  // membentuk polinomial dari koefisien tertentu <b>public Polinomial(double[]</b> coefficients)  // membentuk polinomial dari derajat tertentu dengan koefisien 1 // dan konstanta 0 <b>public Polinomial(int</b> degree)</pre>
<b>Class Methods</b>
<pre>// mengatur nilai koefisien polinomial <b>public void setCoefficients(double[]</b> coefficients)  // menginterpolasi berdasarkan titik-titik yang disediakan dalam // bentuk objek Matriks <b>public void interpolate(Matriks</b> points)  // menghitung nilai polinomial pada x tertentu <b>public double calculate(double</b> x)  // memperbarui koefisien polinomial <b>public void setCoefficients(double[]</b> coefficients)  // melakukan interpolasi terhadap titik-titik yang diberikan</pre>

```
public void interpolate(Matriks points)
// menampilkan koefisien polinomial
public void printCoefficients()
// memperbarui derajat polinomial
private void updateDegree()
```

### 3.5 LinearRegression.java

<b>Attribute</b>
-
<b>Constructor</b>
-
<b>Class Methods</b>
// menambahkan quadratic features dalam metode Quadratic Linear Regression sebelum diolah dengan Normal Equation public Matriks addQuadratic(Matriks X) // memperhitungan Multiple Linear Regression dengan Normal Equation public Matriks normalEquation(Matriks X, Matriks Y) // memprediksi nilai Y berdasarkan perhitungan public double predict(Matriks XNew, Matriks b)

### 3.6 BicubicSplineInterpolation.java

<b>Attribute</b>
// progres eksekusi fungsi resizeImage bernilai 0 sampai 1 private static double resizeProgress
<b>Constructor</b>

**Class Methods**

```

// Menghasilkan interpolasi bicubic spline dari nilai-nilai f pada titik (x, y)
public double BicubicSplineInterpolate(Matriks fValue, double x, double y)

// Menghasilkan gambar yang telah di-resize dengan skala lebar x dan skala tinggi y dengan metode bicubic spline interpolation
public BufferedImage resizeImage(BufferedImage image, double scale_x, double scale_y)

// Melakukan pre-komputasi nilai koefisien a untuk setiap blok 4x4 untuk mengefisienkan performa
private double[][][] preprocessAMatriks(Matriks imageMatriks, Matriks XD)

// Mencari nilai f citra gambar pada titik (x, y), yaitu indeks gambar yang telah dinormalisasi ke range [0..1], dengan interpolasi bicubic spline
private int findFValue(Matriks imageMatriks, int row, int col, double x, double y, Matriks XD, double[][][] a)

// Melakukan interpolasi bicubic spline untuk mengisi nilai pada gambar yang di-resize
private Matriks imageInterpolate(Matriks imageMatriks, double scale_x, double scale_y)

// Mencari nilai koefisien interpolasi a yang lebih akurat dengan matriks X, matriks D, dan nilai citra gambar
private Matriks findImprovedAMatrix(Matriks fValue, Matriks XD)

```

```
// menghitung koefisien interpolasi dengan menyelesaikan sistem  
persamaan linier  
private double[] findAMatriks(Matriks fValue)  
  
// membuat matriks X yang digunakan dalam bicubic spline  
interpolation, berisi kombinasi pangkat x dan y  
private Matriks matriksX()  
  
// membuat matriks D yang digunakan dalam bicubic spline  
interpolation, berisi pengali nilai f dan turunan parsialnya yang  
didekati dengan nilai citra gambar disekitarnya  
private Matriks matriksD()  
  
// Mengembalikan variabel static resizeProgress  
public static double getResizeProgress()  
  
// Mengganti nilai variabel static resizeProgress  
public static void setResizeProgress(double progress)
```

### 3.7 App.java

#### Attribute

```
// menyimpan objek dari kelas scene  
private static Scene scene
```

#### Constructor

```
-
```

#### Class Methods

```
// menginisialisasi antarmuka pengguna
public void start(Stage stage) throws IOException
// memuat file FXML baru dan menggantikan elemen root dari scene
dengan elemen baru yang dimuat
static void setRoot(String fxml) throws IOException
// membaca dan mengonversi file FXML menjadi objek Parent
private static Parent loadFXML(String fxml) throws IOException
// menjalankan aplikasi
public static void main(String[] args)
```

### 3.8 Bicubic Spline Controller

#### Attribute

```
// menampilkan dan menerima input konfigurasi matriks dari
pengguna
@FXML
TextArea inputKonfigurasi = new TextArea()
// menerima input nilai x
@FXML
TextField inputXBebas = new TextField()
// menerima input nilai y
@FXML
TextField inputYBebas = new TextField()
// memulai proses interpolasi bicubic spline
@FXML
Button interpolasiBSButton = new Button()
// menampilkan pesan peringatan atau kesalahan
@FXML
Text alertMsg = new Text()
// menampilkan hasil interpolasi di dalam GUI
@FXML
TextFlow bicubicSplineTextFlow = new TextFlow()
```

```
// menyimpan referensi ke file input yang dipilih oleh pengguna
private File inputFile
// membaca data dari file input
private Scanner scanner
// memfasilitasi pemilihan file dalam GUI
FileChooser fileChooser = new FileChooser()
// menyimpan status apakah interpolasi bicubic telah dilakukan
atau belum
private boolean bicubicSolved
// menyimpan hasil interpolasi dalam bentuk string untuk ekspor ke
file
private String outputString
// menyimpan referensi ke file output yang dipilih pengguna untuk
menyimpan hasil interpolasi
private File outputFile
```

### Constructor

-

### Class Methods

```
// menginisialisasi nilai awal dari atribut
public void initialize()
// kembali ke main menu
private void switchToMainMenu() throws IOException
// export text file pada lokasi sesuai input pengguna
private void exportFile() throws IOException
// buka file matriks konfigurasi dan input x y bicubic spline dan
memasukkan ke text field dan text area
private void chooseFile() throws IOException
// interpolasi bicubic spline
private void interpolasiBicubicSpline()
```

### 3.9 DeterminanController.java

#### Attribute

```
// menampilkan dan menerima input matriks dari pengguna
@FXML
TextArea inputMatriks = new TextArea()
// memasukkan jumlah baris/kolom matriks
@FXML
TextField barisInput = new TextField()
// menampilkan pesan kesalahan atau peringatan kepada pengguna
@FXML
Text alertMsg = new Text()
// memilih metode perhitungan determinan
@FXML
ToggleGroup MetodeDeterminan = new ToggleGroup()
// memilih metode ekspansi kofaktor
@FXML
RadioButton selectEkspansiKofaktor = new RadioButton()
// memilih metode matriks segitiga
@FXML
RadioButton selectMatriksSegitiga = new RadioButton()
// menghitung determinan berdasarkan input pengguna
@FXML
Button determinanButton = new Button()
// menampilkan solusi atau hasil perhitungan determinan di GUI
@FXML
TextFlow solutionTextFlow = new TextFlow()
// file yang dipilih pengguna untuk di-Load sebagai input matriks
private File inputFile
// file untuk menyimpan hasil output
private File outputFile
// scanner untuk membaca file input
```

```
private Scanner scanner  
// menampilkan dialog file chooser  
FileChooser fileChooser = new FileChooser()  
// menyimpan status apakah determinan telah dihitung atau belum  
private boolean determinanSolved  
// menyimpan hasil output dalam bentuk string untuk dieksport ke  
file  
private String outputString
```

#### Constructor

-

#### Class Methods

```
// menginisialisasi atribut  
public void initialize()  
// export text file pada lokasi sesuai input pengguna  
private void exportFile() throws IOException  
// buka file matriks yang akan dicari determinannya dan memasukkan  
ke text field dan text area  
private void chooseFile() throws IOException  
// kembali ke main menu  
private void switchToMainMenu() throws IOException  
// menyelesaikan SPL dengan metode yang dipilih pada GUI  
private void findDeterminan()
```

### 3.10 FileHandler.java

#### Attribute

-

#### Constructor

-

### Class Methods

```
// membaca isi file matriks dari file yang diinputkan pengguna
public static void readMatriks(String[] args)
// menulis matriks 2D ke file bernama output.txt
public static void writeMatriks()
// menyimpan teks yang diberikan ke file yang dipilih oleh
pengguna
public void saveTextToFile(String content, File file)
```

### 3.11 InterpolasiPolinomialController.java

#### Attribute

```
// input jumlah titik (n) yang akan digunakan untuk interpolasi
@FXML
TextField jumlahTitikInput = new TextField()
// input area untuk memasukkan koordinat titik-titik yang akan
digunakan untuk interpolasi
@FXML
TextArea inputTitikList = new TextArea()
// menampilkan pesan peringatan atau kesalahan kepada pengguna
@FXML
Text alertMsg = new Text()
// tombol untuk menampilkan interpolasi
@FXML
Button interpolasiButton = new Button()
// tombol untuk menampilkan polinom
@FXML
TextFlow polinomTextFlow = new TextFlow()
// menambahkan text
@FXML
TextField inputX = new TextField()
```

```
// tombol untuk menghitung fungsi
@FXML
Button hitungFungsiButton = new Button()
// text untuk menghitung fungsi
@FXML
TextField fungsiTextField = new TextField()
// object Polinomial
private Polinomial polinomial = new Polinomial()
// menandai bahwa interpolasi sudah dilakukan
private boolean interpolated
// menandai bahwa kalkulasi variabel bebas pada polinom sudah
dilakukan
private boolean calculated
// nilai x terkecil
private double xmin
// nilai x terbesar
private double xmax
// file text masukan
private File inputFile
// file text keluaran
private File outputFile
// object untuk membaca file
private Scanner scanner
// object untuk memilih file
FileChooser fileChooser = new FileChooser()
// string fungsi polinomial yang ditulis pada file
private String outputPolinomString;
// string nilai fungsi polinomial pada x yang ditulis pada file
private String outputFungsiString;
```

### Constructor

-

**Class Methods**

```
//inisialisasi status interpolasi
public void initialize()
//export text file pada lokasi sesuai input pengguna
private void exportFile() throws IOException
//buka file input interpolasi dan memasukkan ke text field dan
text area
private void chooseFile() throws IOException
//kembali ke main menu
private void switchToMainMenu() throws IOException
//interpolasi polinomial
private void interpolasiPolinomial()
//hitung nilai fungsi polinomial pada x
private void hitungFungsi()
```

### 3.12 MainMenuController.java

**Attribute**

-

**Constructor**

-

**Class Methods**

```
//mengubah tampilan aplikasi ke halaman "Sistem Persamaan Linier"
private void switchToSistemPersamaanLinier() throws IOException
//navigasi ke halaman "determinan"
private void switchToDeterminan() throws IOException
//mengubah root aplikasi ke halaman yang berfokus pada operasi
matriks balikan
private void switchToMatriksBalikan() throws IOException
```

```
// navigasi ke halaman yang menangani Interpolasi Polinomial  
private void switchToInterpolasiPolinomial() throws IOException  
// pindah ke halaman yang menangani Regresi Berganda  
private void switchToRegresiBerganda() throws IOException  
// navigasi ke halaman yang menangani Bicubic Spline  
private void switchToBicubicSpline() throws IOException  
// navigasi ke halaman yang menangani Image Resizer  
private void switchToImageResizer() throws IOException  
// menutup aplikasi  
private void keluar()
```

### 3.13 MatriksBalikanController.java

#### Attribute

```
// memasukkan jumlah baris (yang sama dengan kolom) dari matriks  
// yang akan dicari balikannya  
@FXML  
TextField barisInput = new TextField()  
// memasukkan elemen-elemen matriks secara manual  
@FXML  
TextArea inputMatriks = new TextArea()  
// memulai proses perhitungan balikan matriks  
@FXML  
Button balikanButton = new Button()  
// pilihan metode balikan matriks  
@FXML  
ToggleGroup MetodeBalikan = new ToggleGroup()  
// memilih metode OBE (Operasi Baris Elementer) untuk menghitung  
// balikan matriks  
@FXML  
RadioButton selectOBE = new RadioButton()  
// memilih metode Adjoin untuk menghitung balikan matriks
```

```
@FXML  
RadioButton selectMatriksAdjoin = new RadioButton()  
// hasil balikan matriks akan ditampilkan setelah perhitungan  
selesai  
  
@FXML  
TextFlow solutionTextFlow = new TextFlow()  
// menampilkan pesan peringatan atau error kepada pengguna jika  
terjadi kesalahan dalam input atau perhitungan  
  
@FXML  
Text alertMsg = new Text()  
//inisialisasi input file  
private File inputFile  
//inisialisasi output file  
private File outputFile  
//inisialisasi scan file  
private Scanner scanner  
//inisialisasi pemilihan file  
FileChooser fileChooser = new FileChooser()  
// true apabila sudah terselesaikan  
private boolean balikanSolved  
// output berupa string  
private String outputString
```

### Constructor

-

### Class Methods

```
//inisialisasi metode balikan  
public void initialize()  
// export text file pada lokasi sesuai input pengguna  
private void exportFile() throws IOException  
// buka file matriks yang akan dicari balikannya dan memasukkan ke
```

```
text field dan text area  
private void chooseFile() throws IOException  
// ubah ke main menu  
private void switchToMainMenu() throws IOException  
// mencari balikan matriks dengan metode yang dipilih pada GUI  
private void findBalikan()
```

### 3.14 RegresiBergandaController.java

#### Attribute

```
// menampung input jumlah peubah bebas (variabel X)  
@FXML  
TextField jumlahPeubahInput = new TextField()  
// menampung input jumlah sampel (data X dan Y)  
@FXML  
TextField jumlahSampelInput = new TextField()  
// memasukkan nilai X (peubah bebas)  
@FXML  
TextArea inputX = new TextArea()  
// memasukkan nilai Y (variabel terikat)  
@FXML  
TextArea inputY = new TextArea()  
// menampilkan error atau notifikasi di GUI  
@FXML  
Text alertMsgRegresi = new Text()  
// tombol untuk menjalankan proses regresi  
@FXML  
Button regresiButton = new Button()  
// pilihan jenis regresi linier  
@FXML  
RadioButton selectLinier = new RadioButton()  
// pilihan jenis regresi kuadratik
```

```
@FXML  
RadioButton selectKuadratik = new RadioButton()  
// mengeLompokkan pilihan antara regresi Linier atau kuadratik  
  
@FXML  
ToggleGroup jenisRegresi = new ToggleGroup()  
// mmemasukkan nilai variabel bebas yang akan ditaksir (prediksi)  
  
@FXML  
TextArea inputVariabelBebas = new TextArea()  
// alert untuk taksiran (prediksi)  
  
@FXML  
Text alertMsgTaksiran = new Text()  
// memproses taksiran berdasarkan model regresi  
  
@FXML  
Button taksirButton = new Button()  
// menampilkan hasil persamaan regresi  
  
@FXML  
TextFlow regresiTextFlow = new TextFlow()  
// menampilkan hasil prediksi nilai variabel terikat  
  
@FXML  
TextFlow taksiranTextFlow = new TextFlow()  
// menjalankan algoritma regresi linier atau kuadratik  
  
private LinearRegression regressor = new LinearRegression()  
// matriks hasil regresi yang menyimpan parameter model  
  
private Matriks b  
// status apakah regresi telah dijalankan  
  
private boolean regressed  
// status apakah taksiran (prediksi) telah dilakukan  
  
private boolean predicted  
// jumlah peubah bebas (X)  
  
private int nPeubah  
// jumlah sampel (baris data X dan Y)  
  
private int mSampel
```

```
// tipe regresi yang dipilih (Linier atau kuadratik)
private RegresiType regresiType
// file input dan output untuk membaca data dan menyimpan hasil
regresi
private File inputFile
// file input dan output untuk membaca data dan menyimpan hasil
regresi
private File outputFile
// membaca data dari file input
private Scanner scanner
// memilih file input/output
FileChooser fileChooser = new FileChooser()
// menyimpan string hasil regresi untuk disimpan ke file
private String outputRegresiString
// menyimpan string hasil taksiran untuk disimpan ke file
private String outputTaksirString
```

### Constructor

-

### Class Methods

```
//inisialisasi regresi
public void initialize()
// export text file pada lokasi sesuai input pengguna
private void exportFile() throws IOException
// buka file sampel regresi dan memasukkan ke text field dan text
area
private void chooseFile() throws IOException
//
private void switchToMainMenu() throws IOException
// melakukan regresi
private void regresi()
```

```
// melakukan taksiran nilai Y berdasarkan inputan user
private void taksir()
// menampilkan regresi linier pada text flow
private void displayLinearRegression(Matriks b)
// menampilkan regresi kuadratik pada text flow
private void displayQuadraticRegression(Matriks b)
// mengecek apakah matriks memiliki satu solusi
private boolean hasOneSolution(Matriks solution)
```

### 3.15 SPLController.java

#### Attribute

```
// menerima input berupa matriks dari pengguna
@FXML
TextArea inputMatriks = new TextArea()
// mengelompokkan opsi metode yang akan digunakan untuk
menyelesaikan SPL
@FXML
ToggleGroup MetodeSPL = new ToggleGroup()
// memilih metode Eliminasi Gauss
@FXML
RadioButton selectEliminasiGauss = new RadioButton()
// memilih metode Eliminasi Gauss-Jordan
@FXML
RadioButton selectEliminasiGaussJordan = new RadioButton()
// memilih metode Matriks Balikan
@FXML
RadioButton selectMatriksBalikan = new RadioButton()
// memilih metode Kaidah Cramer
@FXML
RadioButton selectKaidahCramer = new RadioButton()
// menampilkan pesan peringatan atau notifikasi kepada pengguna
```

```
@FXML  
Text alertMsg = new Text()  
// menampilkan solusi SPL dalam bentuk teks di antarmuka pengguna  
@FXML  
TextFlow solutionTextFlow = new TextFlow()  
// menerima input jumlah baris matriks dari pengguna  
@FXML  
TextField barisInput = new TextField()  
// menerima input jumlah kolom matriks dari pengguna  
@FXML  
TextField kolomInput = new TextField()  
// menyimpan referensi ke file input yang dipilih oleh pengguna  
melalui file chooser  
private File inputFile  
// menyimpan referensi ke file output tempat hasil penyelesaian  
SPL disimpan  
private File outputFile  
// membaca data dari input file  
private Scanner scanner  
// memilih file input dan output dengan antarmuka grafis  
FileChooser fileChooser = new FileChooser()  
// menyimpan hasil penyelesaian SPL dalam bentuk string  
private String outputString  
// menyimpan status apakah SPL sudah berhasil diselesaikan atau  
belum  
private boolean splSolved
```

### Constructor

-

### Class Methods

```
// inisialisasi SPL
```

```
public void initialize()
// export text file pada lokasi sesuai input pengguna
private void exportFile() throws IOException
// buka file matriks SPL dan memasukkan ke text field dan text
area
private void chooseFile() throws IOException
// kembali ke main menu
private void switchToMainMenu() throws IOException
// menyelesaikan SPL dengan metode yang dipilih pada GUI
private void solveSPL() throws IOException
// menghapus kolom-kolom yang berisi nol dimulai dari kolom
startCol
private Matriks removeZeroCols(Matriks M, int startCol)
// mengecek apakah kolom matriks M semuanya bernilai 0
private boolean checkZeroCol(Matriks M, int col)
// mengecek apakah baris matriks M semuanya bernilai 0
private boolean checkZeroRow(Matriks M, int row)
```

### 3.16 TextFlowHandler.java

Attribute
-
Constructor
-
Class Methods
// membaca file menjadi string public String textFlowToString(TextFlow tf)

### 3.17 ImageResizerController.java

#### Attribute

```
// menampilkan gambar asli sebelum diubah ukurannya
@FXML
ImageView oldImageView = new ImageView()
// menampilkan gambar setelah diubah ukurannya
@FXML
ImageView resizedImageView = new ImageView()
// menampilkan ukuran gambar asli (width x height)
@FXML
Text oldSizeText = new Text()
// menampilkan ukuran gambar setelah diubah ukurannya
@FXML
Text newSizeText = new Text()
// peringatan jika terjadi kesalahan atau masalah saat mengubah
ukuran gambar
@FXML
Text resizeAlertMsg = new Text()
// peringatan jika terjadi kesalahan atau masalah saat mengekspor
gambar
@FXML
Text exportAlertMsg = new Text()
// input teks untuk menentukan skala lebar gambar saat diubah
ukurannya
@FXML
TextField scaleWidthInput = new TextField()
// input teks untuk menentukan skala tinggi gambar saat diubah
ukurannya
@FXML
TextField scaleHeightInput = new TextField()
// gambar asli yang diambil dari file yang dipilih oleh pengguna
```

```
private BufferedImage oldImage
// gambar yang telah diubah ukurannya
private BufferedImage resizedImage
// menandai apakah gambar telah dimuat atau belum (untuk memeriksa
// validitas operasi)
private boolean imageLoaded
// menandai apakah gambar telah diubah ukurannya atau belum (untuk
// memeriksa validitas operasi ekspor)
private boolean imageResized
// ukuran standar untuk menampilkan gambar asli
private final double oldImageViewWidth = 300
// ukuran standar untuk menampilkan gambar asli
private final double oldImageViewHeight = 240
```

### Constructor

-

### Class Methods

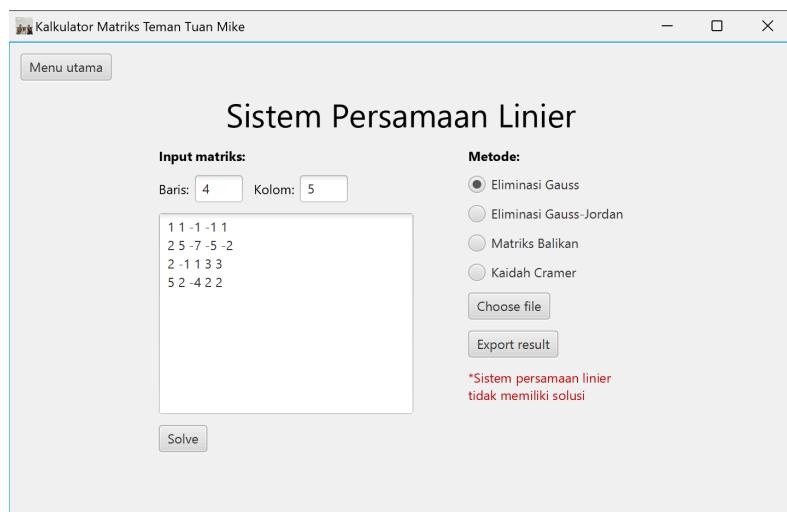
```
// menginisialisasi fungsi image resizer
public void initialize()
// mengembalikan ke main menu
private void switchToMainMenu() throws IOException
// memasukkan foto yang hendak diresize
private void chooseImage()
// meresize foto
private void resize()
// export gambar yang sudah diubah ukurannya menjadi file gambar
baru
private void export()
// mengatur tampilan gambar asli di oldImageView dengan menjaga
// rasio aspek dan menetapkan ukuran tampilan yang telah ditentukan
// sebelumnya
```

```
private void displayOldImage(Image image)
// menjaga rasio aspek dan memastikan ukuran tampilan berada dalam
rentang minimum dan maksimum yang diizinkan
private void displayResizedImage(Image image, double frameWidth,
double frameHeight)
```

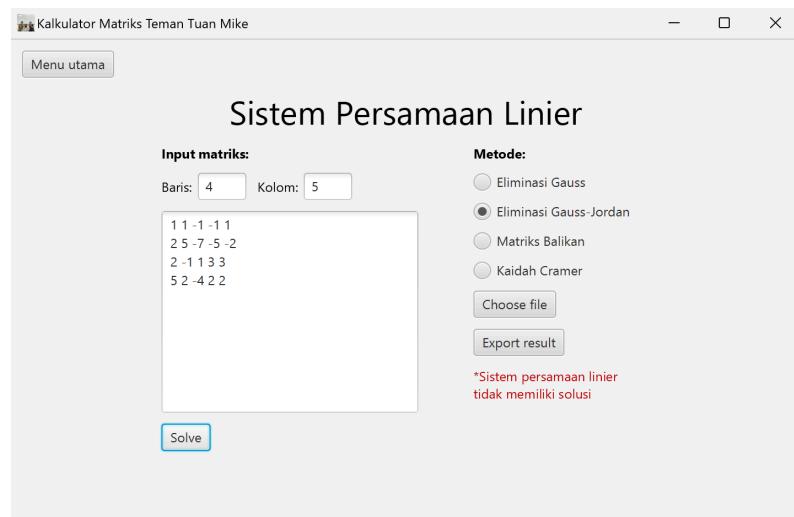
**BAB 4****EKSPERIMEN****4.1. Sistem Persamaan Linier****4.1.1. Kasus 1a**

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

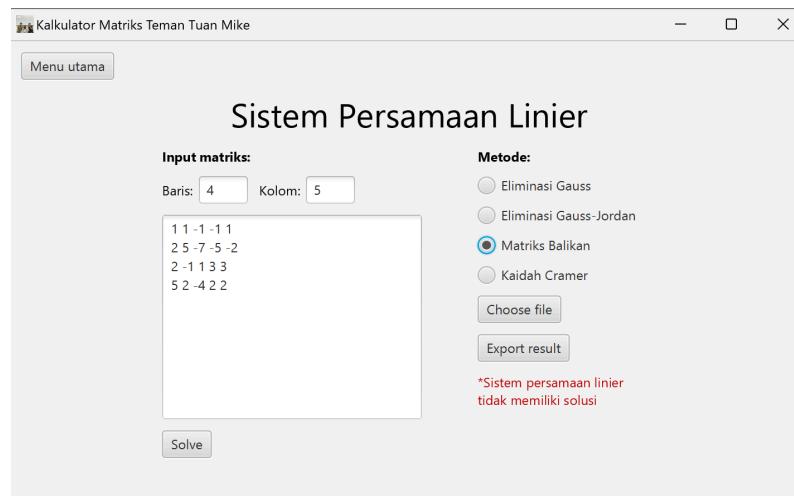
## a. Metode Eliminasi Gauss



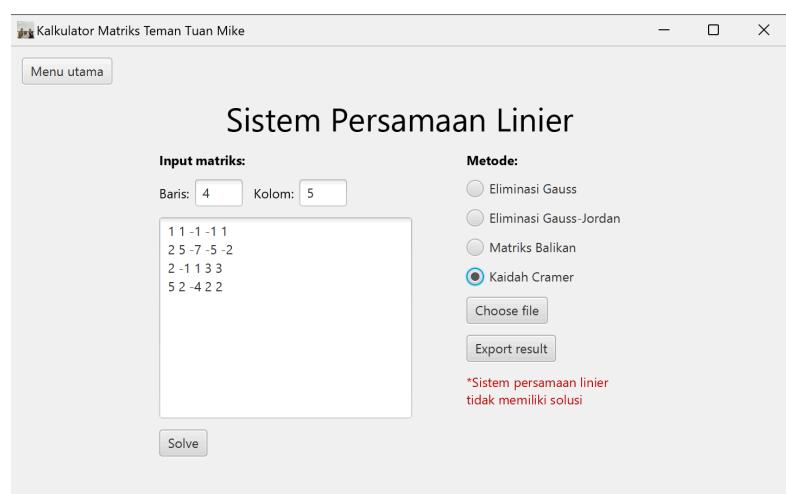
## b. Metode Eliminasi Gauss Jordan



#### c. Metode Matriks Balikan



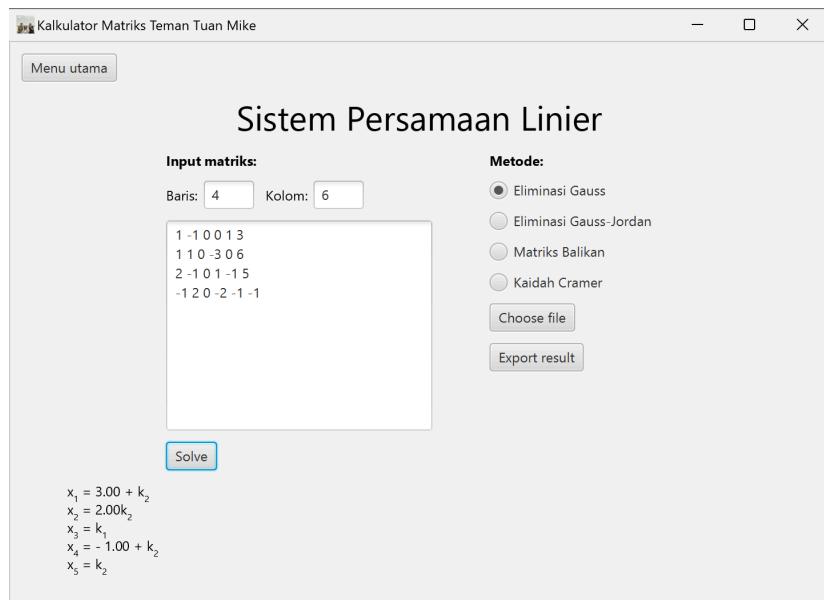
#### d. Kaidah Cramer



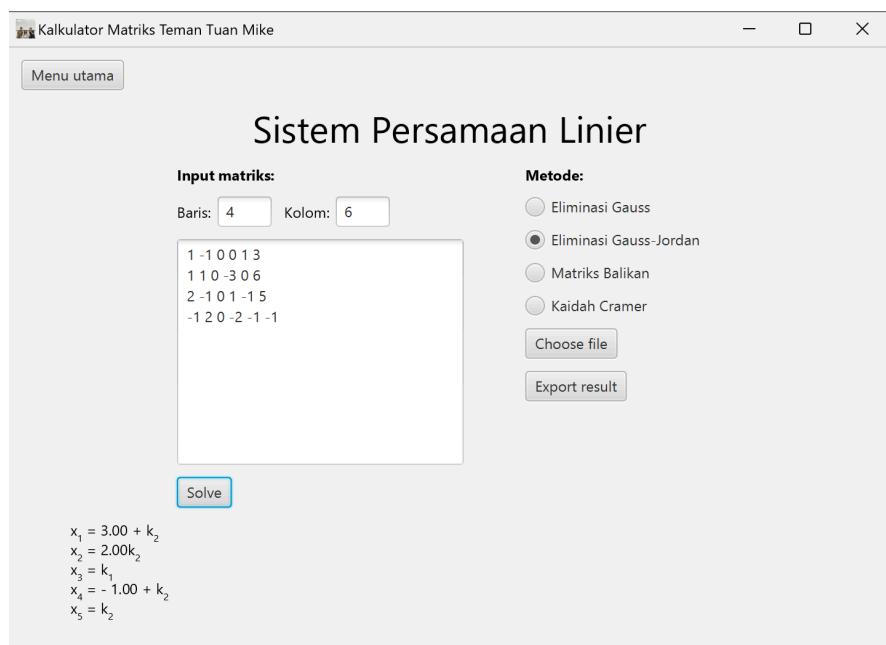
#### 4.1.2. Kasus 1b

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

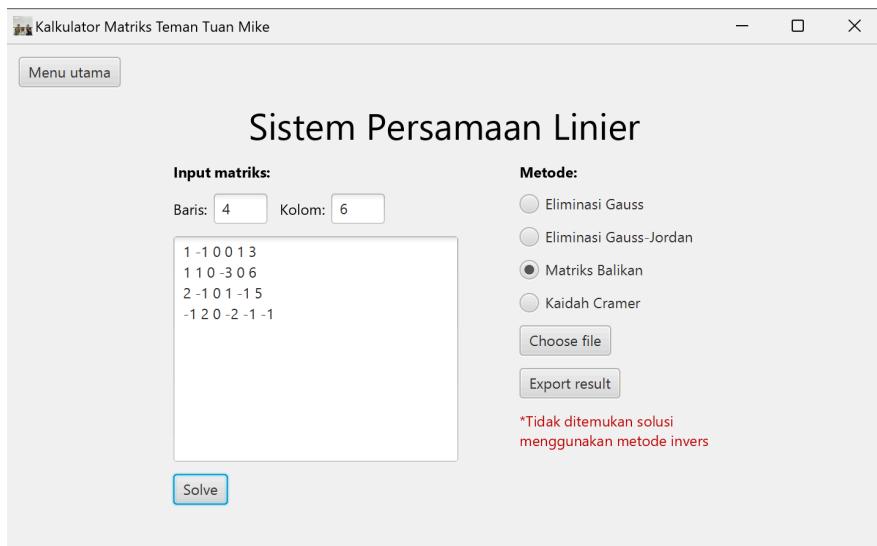
##### a. Metode Eliminasi Gauss



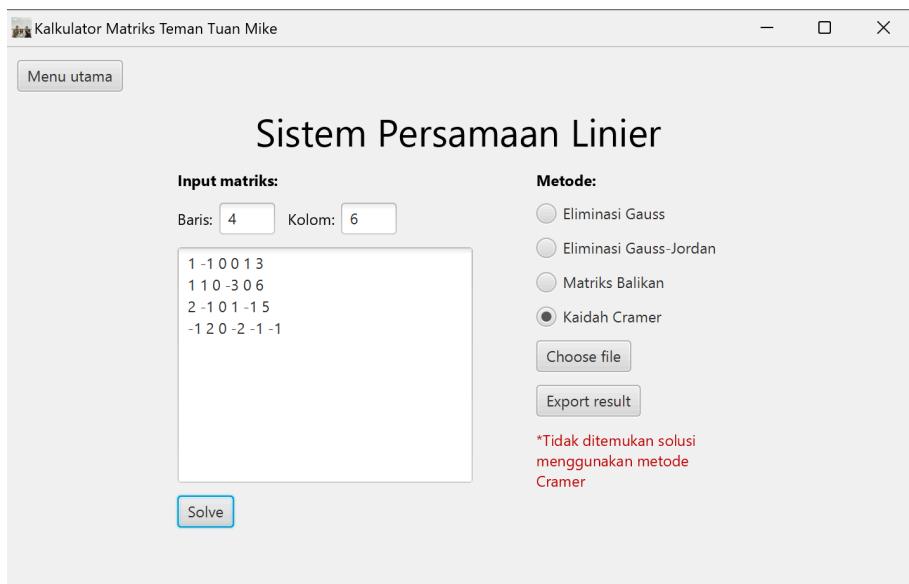
##### b. Metode Eliminasi Gauss Jordan



##### c. Metode Matriks Balikan



#### d. Kaidah Cramer



#### 4.1.3. Kasus 1c

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

#### a. Metode Eliminasi Gauss

Kalkulator Matriks Teman Tuan Mike

Menu utama

### Sistem Persamaan Linier

**Input matriks:**

Baris: 3 Kolom: 7

```

0 1 0 0 1 0 2
0 0 0 1 1 0 -1
0 1 0 0 0 1 1
  
```

**Metode:**

- Eliminasi Gauss
- Eliminasi Gauss-Jordan
- Matriks Balikan
- Kaidah Cramer

**Solve**

$x_1 = k_1$   
 $x_2 = 1.00 - k_3$   
 $x_3 = k_2$   
 $x_4 = -2.00 - k_3$   
 $x_5 = 1.00 + k_3$   
 $x_6 = k_3$

**Choose file**

**Export result**

b. Metode Eliminasi Gauss Jordan

Kalkulator Matriks Teman Tuan Mike

Menu utama

### Sistem Persamaan Linier

**Input matriks:**

Baris: 3 Kolom: 7

```

0 1 0 0 1 0 2
0 0 0 1 1 0 -1
0 1 0 0 0 1 1
  
```

**Metode:**

- Eliminasi Gauss
- Eliminasi Gauss-Jordan
- Matriks Balikan
- Kaidah Cramer

**Solve**

$x_1 = k_1$   
 $x_2 = 1.00 - k_3$   
 $x_3 = k_2$   
 $x_4 = -2.00 - k_3$   
 $x_5 = 1.00 + k_3$   
 $x_6 = k_3$

**Choose file**

**Export result**

c. Metode Matriks Balikan

Kalkulator Matriks Teman Tuan Mike

Menu utama

## Sistem Persamaan Linier

**Input matriks:**

Baris: 3    Kolom: 7

```
0 1 0 0 1 0 2
0 0 0 1 1 0 -1
0 1 0 0 0 1 1
```

**Metode:**

- Eliminasi Gauss
- Eliminasi Gauss-Jordan
- Matriks Balikan
- Kaidah Cramer

**Buttons:**

- Choose file
- Export result

\*Tidak ditemukan solusi menggunakan metode invers

**Solve**

d. Kaidah Cramer

Kalkulator Matriks Teman Tuan Mike

Menu utama

## Sistem Persamaan Linier

**Input matriks:**

Baris: 3    Kolom: 7

```
0 1 0 0 1 0 2
0 0 0 1 1 0 -1
0 1 0 0 0 1 1
```

**Metode:**

- Eliminasi Gauss
- Eliminasi Gauss-Jordan
- Matriks Balikan
- Kaidah Cramer

**Buttons:**

- Choose file
- Export result

\*Tidak ditemukan solusi menggunakan metode Cramer

**Solve**

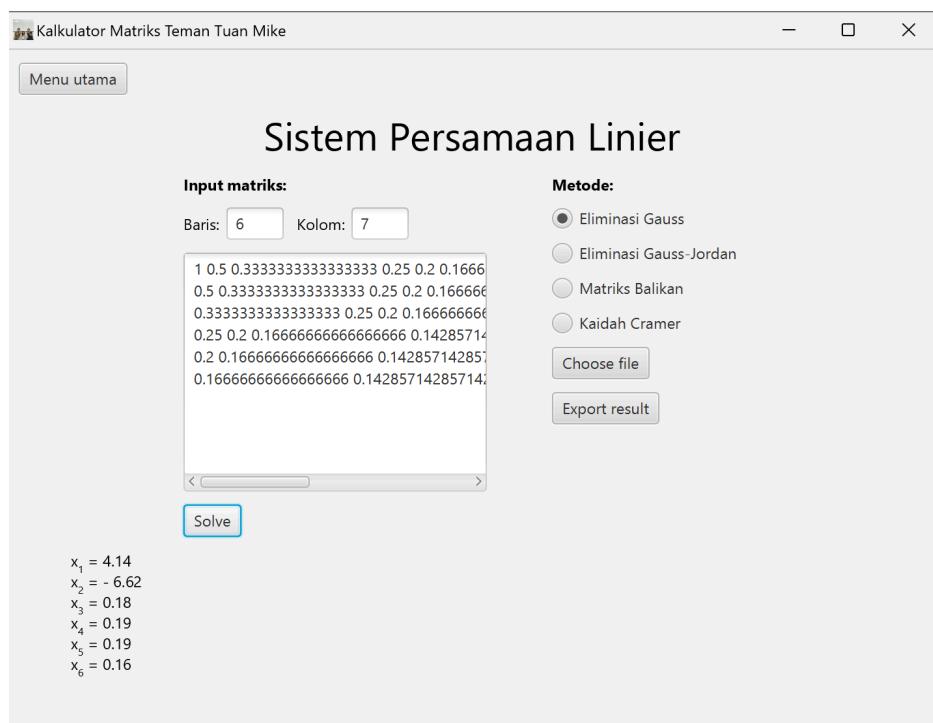
#### 4.1.4. Kasus 1d

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

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

##### 4.1.4.1. Nilai n = 6

###### a. Metode Eliminasi Gauss



###### b. Metode Eliminasi Gauss Jordan

Kalkulator Matriks Teman Tuan Mike

Menu utama

# Sistem Persamaan Linier

**Input matriks:**

Baris: 6    Kolumn: 7

1	0.5	0.3333333333333333	0.25	0.2	0.1666666666666666	0.1428571428571429
0.5	0.3333333333333333	0.25	0.2	0.1666666666666666	0.1428571428571429	0.1428571428571429
0.3333333333333333	0.25	0.2	0.1666666666666666	0.1428571428571429	0.1428571428571429	0.1428571428571429
0.25	0.2	0.1666666666666666	0.1428571428571429	0.1428571428571429	0.1428571428571429	0.1428571428571429
0.2	0.1666666666666666	0.1428571428571429	0.1428571428571429	0.1428571428571429	0.1428571428571429	0.1428571428571429
0.1666666666666666	0.1428571428571429	0.1428571428571429	0.1428571428571429	0.1428571428571429	0.1428571428571429	0.1428571428571429
0.1428571428571429	0.1428571428571429	0.1428571428571429	0.1428571428571429	0.1428571428571429	0.1428571428571429	0.1428571428571429

**Metode:**

Eliminasi Gauss

Eliminasi Gauss-Jordan

Matriks Balikan

Kaidah Cramer

**Solve**

**X<sub>1</sub>** = 4.14  
**X<sub>2</sub>** = - 6.62  
**X<sub>3</sub>** = 0.18  
**X<sub>4</sub>** = 0.19  
**X<sub>5</sub>** = 0.19  
**X<sub>6</sub>** = 0.16

Choose file

Export result

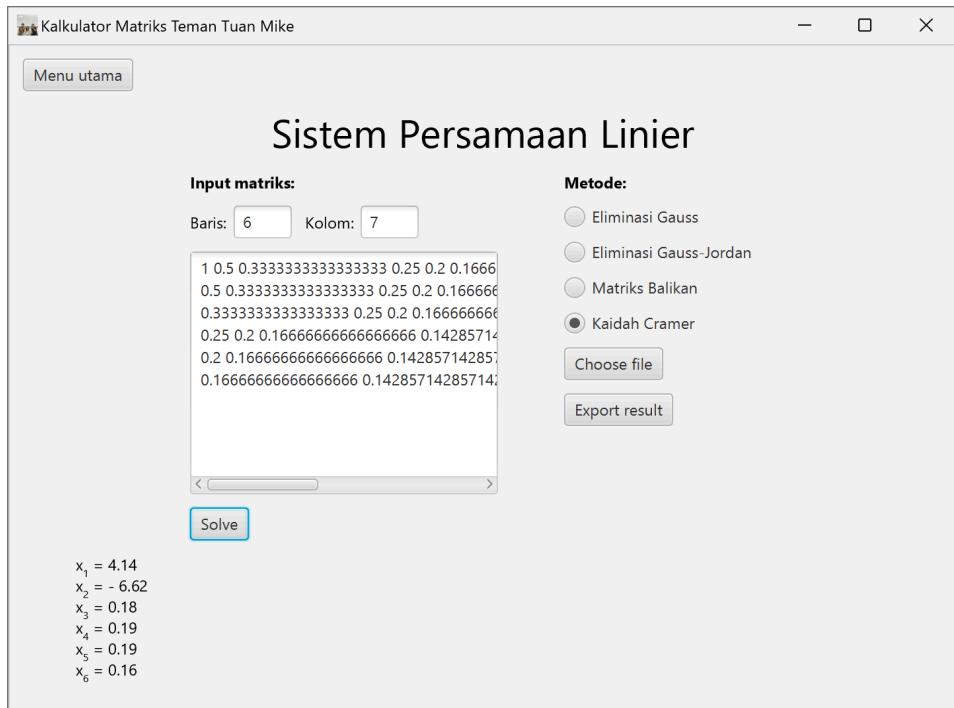
### c. Metode Matriks Balikan

The screenshot shows a window titled "Kalkulator Matriks Teman Tuan Mike". The main title "Sistem Persamaan Linier" is displayed prominently. On the left, there's an "Input matriks:" section with "Baris: 6" and "Kolom: 7" input fields. Below this is a large text area containing a 6x7 matrix of numerical values. To the right, under "Metode:", several solving methods are listed as radio buttons: "Eliminasi Gauss", "Eliminasi Gauss-Jordan" (which is selected), "Matriks Balikan" (selected), and "Kaidah Cramer". There are also "Choose file" and "Export result" buttons. At the bottom left, the solution vector is displayed as:

$$\begin{aligned}x_1 &= 4.14 \\x_2 &= -6.62 \\x_3 &= 0.18 \\x_4 &= 0.19 \\x_5 &= 0.19 \\x_6 &= 0.16\end{aligned}$$

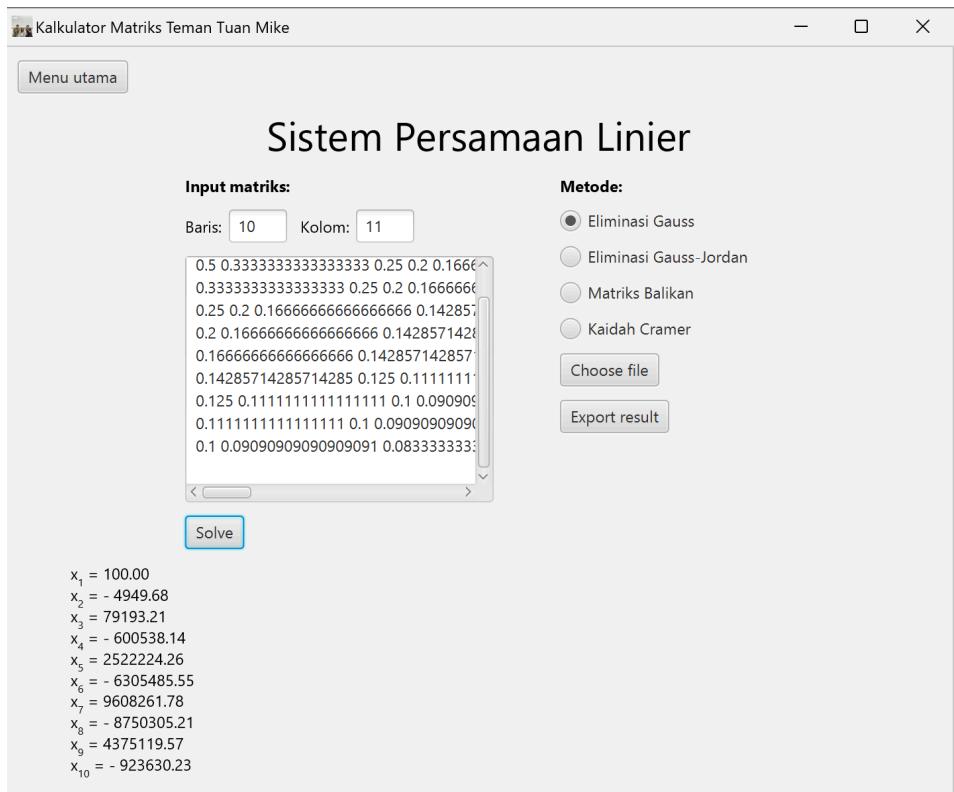
A blue-bordered "Solve" button is located at the bottom left of the matrix input area.

d. Kaidah Cramer



#### 4.1.4.2. Nilai n = 10

##### a. Metode Eliminasi Gauss



b. Metode Eliminasi Gauss Jordan

Kalkulator Matriks Teman Tuan Mike

Menu utama

### Sistem Persamaan Linier

**Input matriks:**

Baris: 10 Kolom: 11

```
0.5 0.3333333333333333 0.25 0.2 0.1666666666666666
0.3333333333333333 0.25 0.2 0.1666666666666666
0.25 0.2 0.1666666666666666 0.1428571428571428
0.2 0.1666666666666666 0.1428571428571428
0.1666666666666666 0.1428571428571428
0.1428571428571428 0.125 0.1111111111111111
0.125 0.1111111111111111 0.1 0.090909090909
0.1111111111111111 0.1 0.0909090909091 0.0833333333333333
0.1 0.09090909090909091 0.0833333333333333
```

**Metode:**

- Eliminasi Gauss
- Eliminasi Gauss-Jordan
- Matriks Balikan
- Kaidah Cramer

**Solve**

$x_1 = 100.00$   
 $x_2 = -4949.68$   
 $x_3 = 79193.21$   
 $x_4 = -600538.14$   
 $x_5 = 252224.26$   
 $x_6 = -6305485.55$   
 $x_7 = 9608261.78$   
 $x_8 = -8750305.21$   
 $x_9 = 4375119.57$   
 $x_{10} = -923630.23$

c. Metode Matriks Balikan

Kalkulator Matriks Teman Tuan Mike

Menu utama

### Sistem Persamaan Linier

**Input matriks:**

Baris: 10 Kolom: 11

```
0.5 0.3333333333333333 0.25 0.2 0.1666666666666666
0.3333333333333333 0.25 0.2 0.1666666666666666
0.25 0.2 0.1666666666666666 0.1428571428571428
0.2 0.1666666666666666 0.1428571428571428
0.1666666666666666 0.1428571428571428
0.1428571428571428 0.125 0.1111111111111111
0.125 0.1111111111111111 0.1 0.090909090909
0.1111111111111111 0.1 0.0909090909091 0.0833333333333333
0.1 0.09090909090909091 0.0833333333333333
```

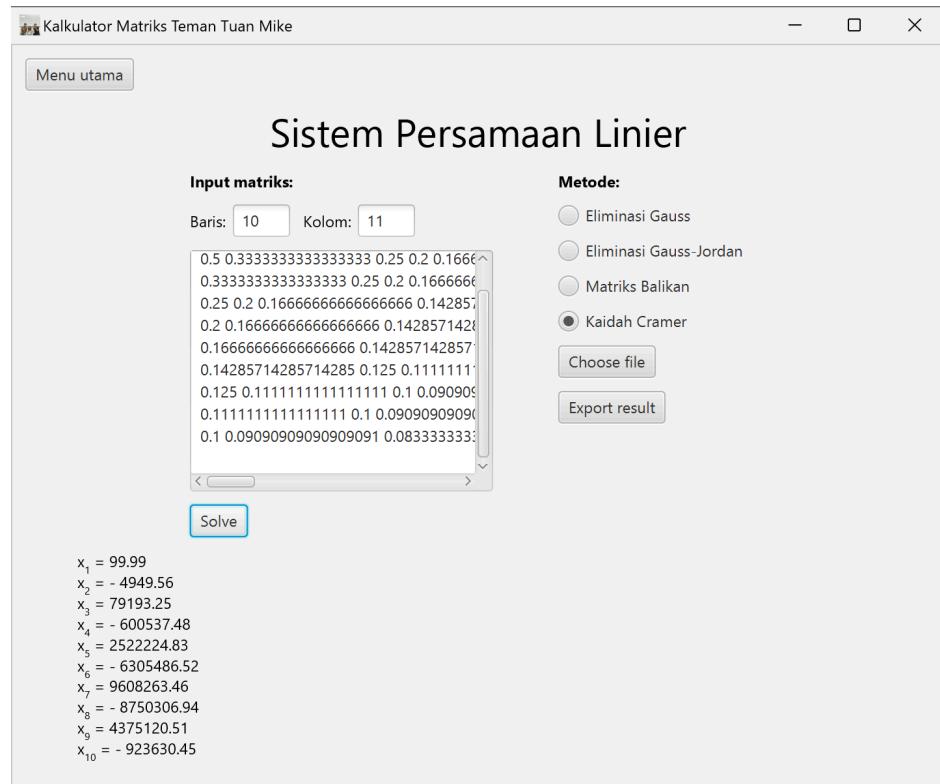
**Metode:**

- Eliminasi Gauss
- Eliminasi Gauss-Jordan
- Matriks Balikan
- Kaidah Cramer

**Solve**

$x_1 = 99.99$   
 $x_2 = -4949.53$   
 $x_3 = 79192.15$   
 $x_4 = -600531.20$   
 $x_5 = 2522205.43$   
 $x_6 = -6305460.31$   
 $x_7 = 9608252.01$   
 $x_8 = -8750318.82$   
 $x_9 = 4375136.16$   
 $x_{10} = -923635.55$

d. Kaidah Cramer



#### 4.1.5. Kasus 2a

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

a. Metode Eliminasi Gauss

Kalkulator Matriks Teman Tuan Mike

Menu utama

### Sistem Persamaan Linier

**Input matriks:**

Baris: 4    Kolom: 5

```

1 -1 2 -1 -1
2 1 -2 -2 -2
-1 2 -4 1 1
3 0 0 -3 -3

```

**Metode:**

- Eliminasi Gauss
- Eliminasi Gauss-Jordan
- Matriks Balikan
- Kaidah Cramer

**Buttons:**

- Choose file
- Export result

**Solve**

$x_1 = -1.00 + k_2$   
 $x_2 = 2.00k_1$   
 $x_3 = k_1$   
 $x_4 = k_2$

b. Metode Eliminasi Gauss Jordan

Kalkulator Matriks Teman Tuan Mike

Menu utama

### Sistem Persamaan Linier

**Input matriks:**

Baris: 4    Kolom: 5

```

1 -1 2 -1 -1
2 1 -2 -2 -2
-1 2 -4 1 1
3 0 0 -3 -3

```

**Metode:**

- Eliminasi Gauss
- Eliminasi Gauss-Jordan
- Matriks Balikan
- Kaidah Cramer

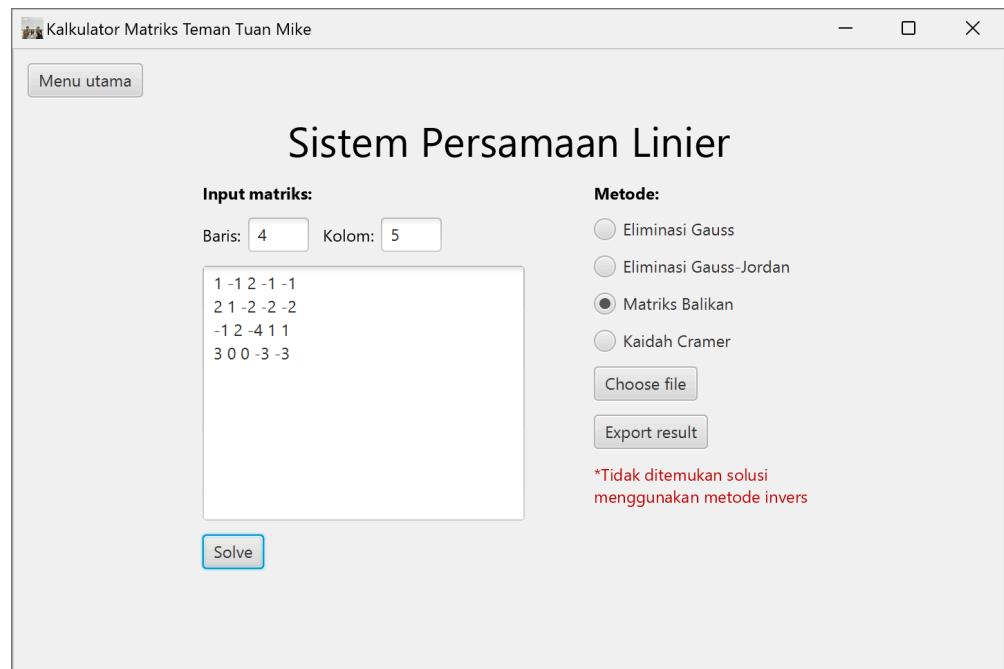
**Buttons:**

- Choose file
- Export result

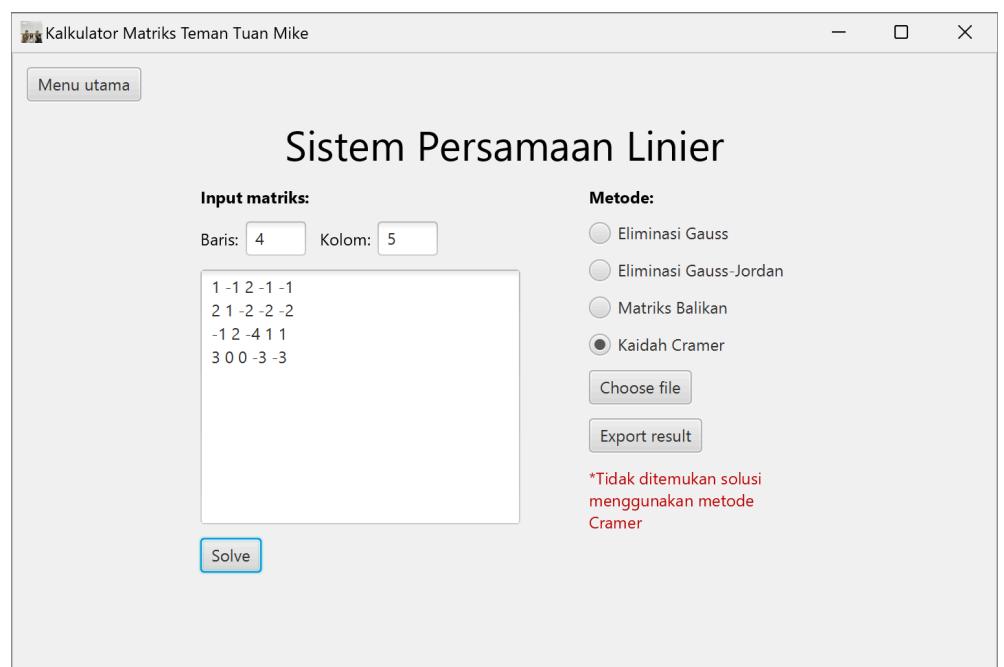
**Solve**

$x_1 = -1.00 + k_2$   
 $x_2 = 2.00k_1$   
 $x_3 = k_1$   
 $x_4 = k_2$

c. Metode Matriks Balikan



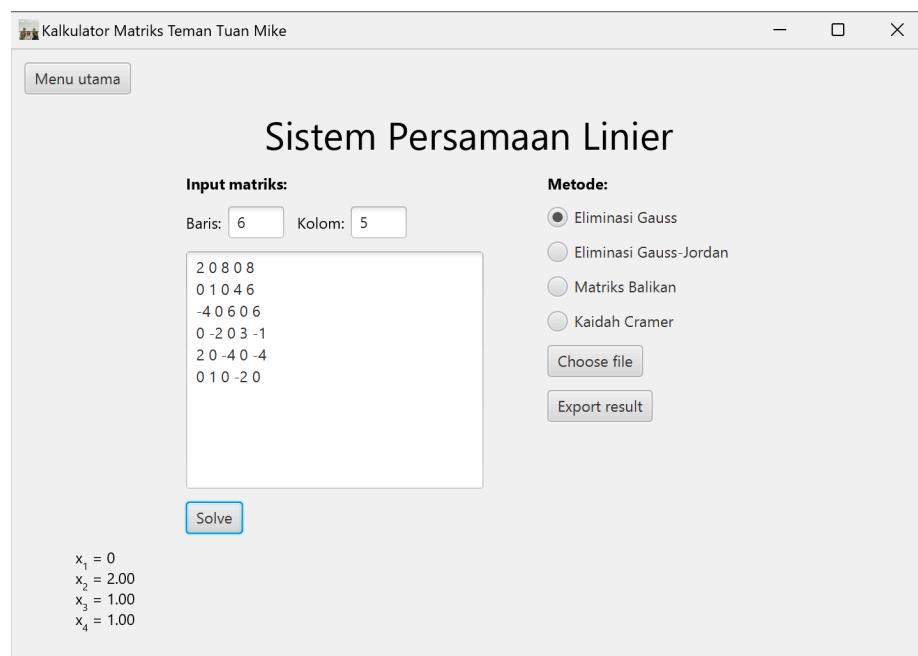
d. Kaidah Cramer



#### 4.1.6. Kasus 2b

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

- a. Metode Eliminasi Gauss



- b. Metode Eliminasi Gauss Jordan

Kalkulator Matriks Teman Tuan Mike

Menu utama

### Sistem Persamaan Linier

**Input matriks:**

Baris: 6 Kolom: 5

```

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

```

**Metode:**

- Eliminasi Gauss
- Eliminasi Gauss-Jordan
- Matriks Balikan
- Kaidah Cramer

**Buttons:**

- Choose file
- Export result

**Solve button:**

**Result:**

$$\begin{aligned}x_1 &= 0 \\x_2 &= 2.00 \\x_3 &= 1.00 \\x_4 &= 1.00\end{aligned}$$

#### c. Metode Matriks Balikan

Kalkulator Matriks Teman Tuan Mike

Menu utama

### Sistem Persamaan Linier

**Input matriks:**

Baris: 6 Kolom: 5

```

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

```

**Metode:**

- Eliminasi Gauss
- Eliminasi Gauss-Jordan
- Matriks Balikan
- Kaidah Cramer

**Buttons:**

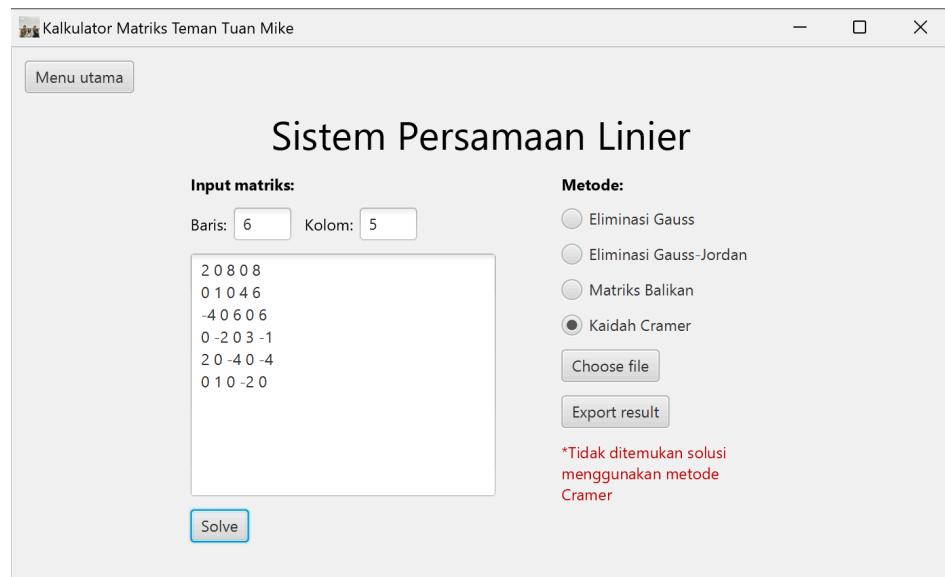
- Choose file
- Export result

**Note:**

\*Tidak ditemukan solusi menggunakan metode invers

**Solve button:**

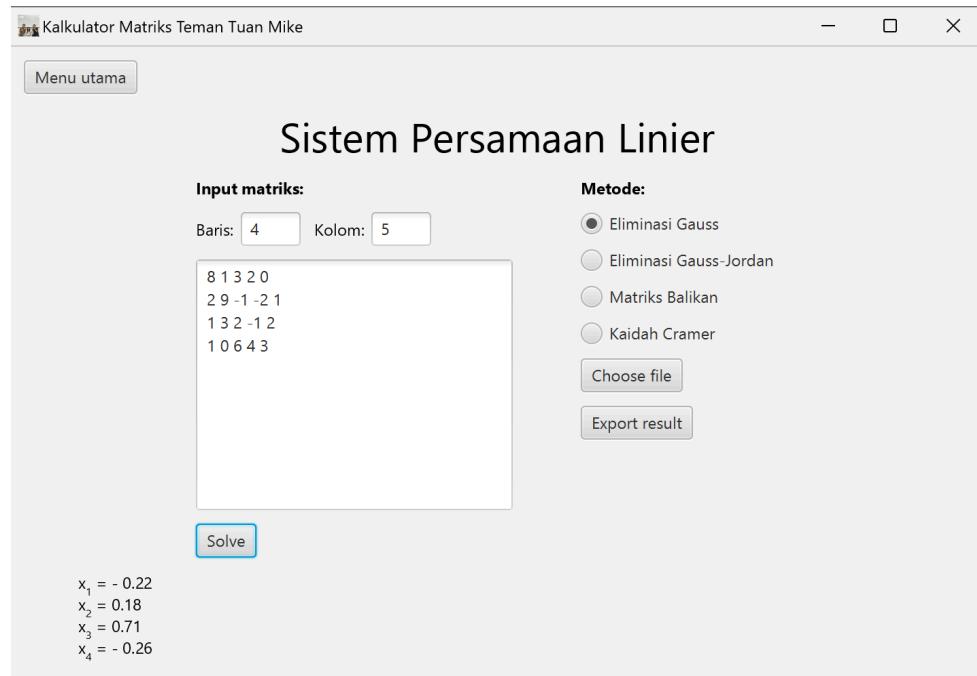
#### d. Kaidah Cramer



#### 4.1.7. Kasus 3a

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

##### a. Metode Eliminasi Gauss



##### b. Metode Eliminasi Gauss Jordan

Kalkulator Matriks Teman Tuan Mike

Menu utama

## Sistem Persamaan Linier

**Input matriks:**

Baris: 4    Kolom: 5

```

8 1 3 2 0
2 9 -1 -2 1
1 3 2 -1 2
1 0 6 4 3

```

**Metode:**

- Eliminasi Gauss
- Eliminasi Gauss-Jordan
- Matriks Balikan
- Kaidah Cramer

**Buttons:**

- Choose file
- Export result

**Solve**

$x_1 = -0.22$   
 $x_2 = 0.18$   
 $x_3 = 0.71$   
 $x_4 = -0.26$

c. Metode Matriks Balikan

Kalkulator Matriks Teman Tuan Mike

Menu utama

## Sistem Persamaan Linier

**Input matriks:**

Baris: 4    Kolom: 5

```

8 1 3 2 0
2 9 -1 -2 1
1 3 2 -1 2
1 0 6 4 3

```

**Metode:**

- Eliminasi Gauss
- Eliminasi Gauss-Jordan
- Matriks Balikan
- Kaidah Cramer

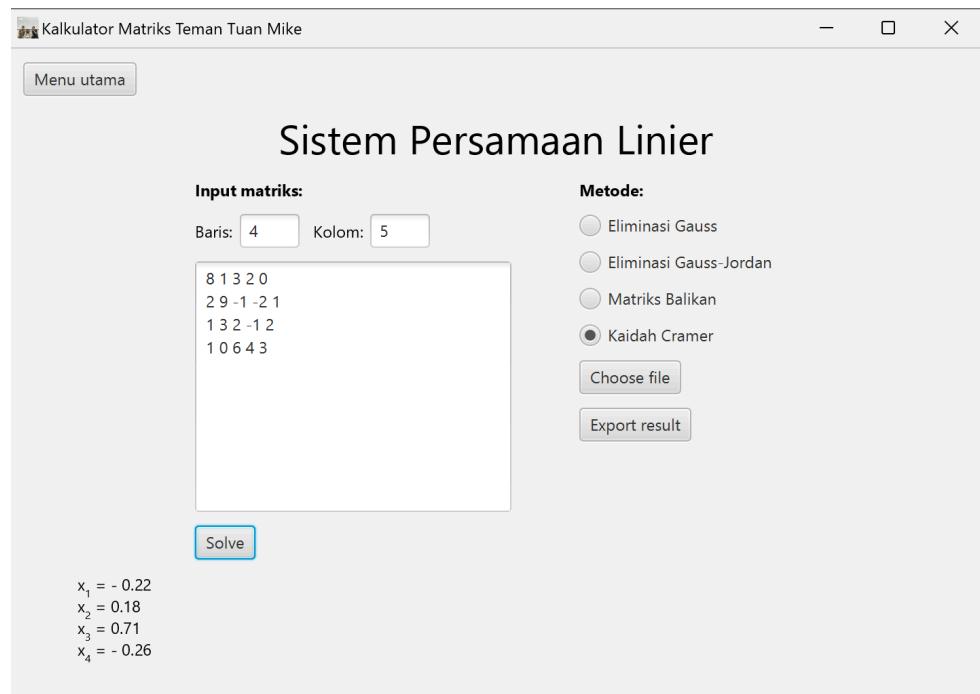
**Buttons:**

- Choose file
- Export result

**Solve**

$x_1 = -0.22$   
 $x_2 = 0.18$   
 $x_3 = 0.71$   
 $x_4 = -0.26$

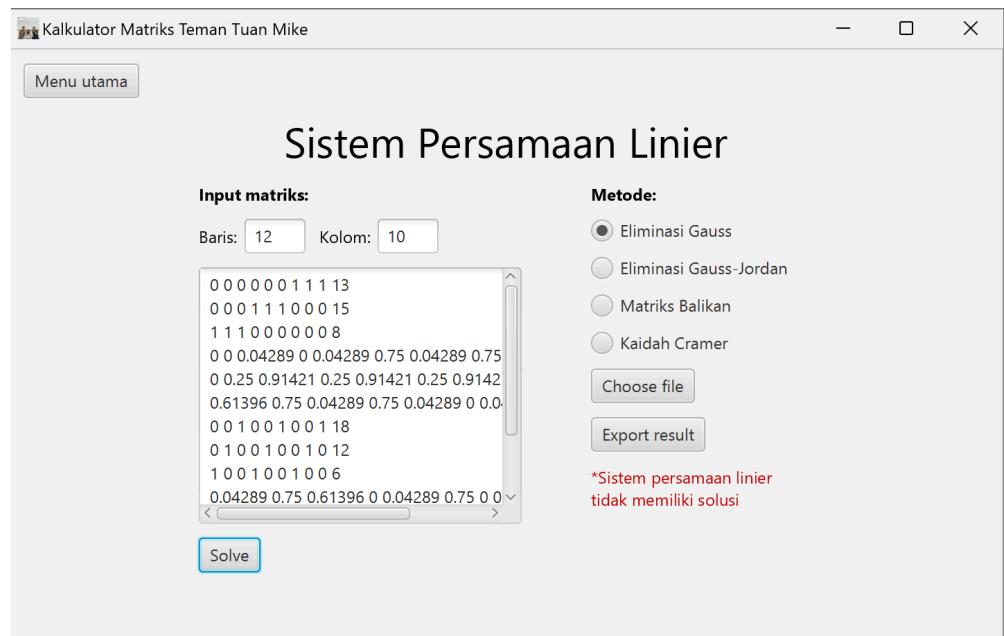
d. Kaidah Cramer



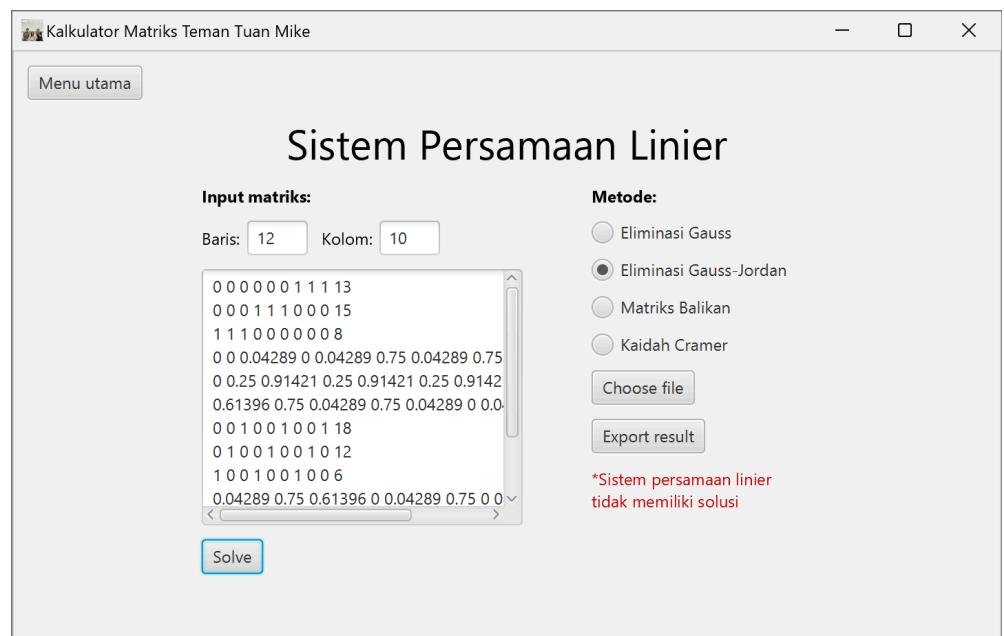
#### 4.1.8. Kasus 3b

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

a. Metode Eliminasi Gauss



b. Metode Eliminasi Gauss Jordan



c. Metode Matriks Balikan

Kalkulator Matriks Teman Tuan Mike

Menu utama

## Sistem Persamaan Linier

**Input matriks:**

Baris: 12 Kolom: 10

```

0 0 0 0 0 0 1 1 1 1 3
0 0 1 1 1 0 0 0 1 5
1 1 1 0 0 0 0 0 0 8
0 0 0 0 4289 0 0 4289 0 75 0 0 4289 0 75
0 0 25 0 91421 0 25 0 91421 0 25 0 9142
0 61396 0 75 0 0 4289 0 75 0 0 4289 0 0 0
0 0 1 0 0 1 0 0 1 1 8
0 1 0 0 1 0 0 1 0 1 2
1 0 0 1 0 0 1 0 0 6
0 0 4289 0 75 0 61396 0 0 0 4289 0 75 0 0 0

```

**Metode:**

- Eliminasi Gauss
- Eliminasi Gauss-Jordan
- Matriks Balikan
- Kaidah Cramer

**Buttons:**

- Choose file
- Export result

\*Tidak ditemukan solusi menggunakan metode invers

**Solve**

d. Kaidah Cramer

Kalkulator Matriks Teman Tuan Mike

Menu utama

## Sistem Persamaan Linier

**Input matriks:**

Baris: 12 Kolom: 10

```

0 0 0 0 0 0 1 1 1 1 3
0 0 1 1 1 0 0 0 1 5
1 1 1 0 0 0 0 0 0 8
0 0 0 0 4289 0 0 4289 0 75 0 0 4289 0 75
0 0 25 0 91421 0 25 0 91421 0 25 0 9142
0 61396 0 75 0 0 4289 0 75 0 0 4289 0 0 0
0 0 1 0 0 1 0 0 1 1 8
0 1 0 0 1 0 0 1 0 1 2
1 0 0 1 0 0 1 0 0 6
0 0 4289 0 75 0 61396 0 0 0 4289 0 75 0 0 0

```

**Metode:**

- Eliminasi Gauss
- Eliminasi Gauss-Jordan
- Matriks Balikan
- Kaidah Cramer

**Buttons:**

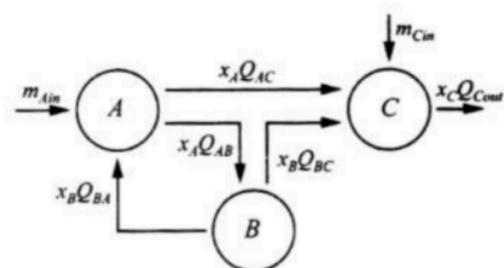
- Choose file
- Export result

\*Tidak ditemukan solusi menggunakan metode Cramer

**Solve**

### 4.1.9. Kasus 4

Diketahui sistem reaktor seperti pada gambar berikut.



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

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

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

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

Dalam kasus ini, akan ditentukan solusi x<sub>A</sub>, x<sub>B</sub>, x<sub>C</sub> dengan menggunakan parameter berikut : Q<sub>AB</sub> = 40, Q<sub>AC</sub> = 80, Q<sub>BA</sub> = 60, Q<sub>BC</sub> = 20 dan Q<sub>Cout</sub> = 150 m<sup>3</sup>/s dan m<sub>Ain</sub> = 1300 dan m<sub>Cin</sub> = 200 mg/s.

#### a. Metode Eliminasi Gauss

Kalkulator Matriks Teman Tuan Mike

Menu utama

### Sistem Persamaan Linier

**Input matriks:**

Baris: 3 Kolom: 4

120	-60	0	1300
-40	80	0	0
-80	-20	150	200

**Metode:**

- Eliminasi Gauss
- Eliminasi Gauss-Jordan
- Matriks Balikan
- Kaidah Cramer

**Solve**

x<sub>1</sub> = 14.44  
x<sub>2</sub> = 7.22  
x<sub>3</sub> = 10.00

#### b. Metode Eliminasi Gauss Jordan

Kalkulator Matriks Teman Tuan Mike

**Sistem Persamaan Linier**

**Input matriks:**  
Baris: 3 Kolom: 4

```

120 -60 0 1300
-40 80 0 0
-80 -20 150 200

```

**Metode:**  
 Eliminasi Gauss  
 Eliminasi Gauss-Jordan  
 Matriks Balikan  
 Kaidah Cramer

**Solve**

$x_1 = 14.44$   
 $x_2 = 7.22$   
 $x_3 = 10.00$

c. Metode Matriks Balikan

Kalkulator Matriks Teman Tuan Mike

**Sistem Persamaan Linier**

**Input matriks:**  
Baris: 3 Kolom: 4

```

120 -60 0 1300
-40 80 0 0
-80 -20 150 200

```

**Metode:**  
 Eliminasi Gauss  
 Eliminasi Gauss-Jordan  
 Matriks Balikan  
 Kaidah Cramer

**Solve**

$x_1 = 14.44$   
 $x_2 = 7.22$   
 $x_3 = 10.00$

d. Kaidah Cramer

Kalkulator Matriks Teman Tuan Mike

Menu utama

## Sistem Persamaan Linier

**Input matriks:**

Baris: 3    Kolom: 4

```
120 -60 0 1300
-40 80 0 0
-80 -20 150 200
```

**Metode:**

- Eliminasi Gauss
- Eliminasi Gauss-Jordan
- Matriks Balikan
- Kaidah Cramer

**Buttons:**

- Choose file
- Export result

**Solve**

$x_1 = 14.44$   
 $x_2 = 7.22$   
 $x_3 = 10.00$

## 4.2. Interpolasi polinomial

### 4.2.1. Kasus 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:

$$x = 0.2 \quad f(x) = ?$$

$$x = 0.55 \quad f(x) = ?$$

$$x = 0.85 \quad f(x) = ?$$

$$x = 1.28 \quad f(x) = ?$$

a.  $x = 0.2$

Kalkulator Matriks Teman Tuan Mike

Menu utama

## Interpolasi Polinomial

**Input titik:**

Jumlah titik:

0.1 0.003
0.3 0.067
0.5 0.148
0.7 0.248
0.9 0.370
1.1 0.518
1.3 0.697

**Input variabel bebas:**

x:

Polinomial:  
 $P(x) = -0.02 + 0.24x^1 + 0.20x^2 + 0.00x^3 + 0.03x^4 + 0.00x^5 - 0.00x^6$

$P(0.2) = 0.0330$

b.  $x = 0.55$

Kalkulator Matriks Teman Tuan Mike

Menu utama

## Interpolasi Polinomial

**Input titik:**

Jumlah titik:

0.1 0.003
0.3 0.067
0.5 0.148
0.7 0.248
0.9 0.370
1.1 0.518
1.3 0.697

**Input variabel bebas:**

x:

Polinomial:  
 $P(x) = -0.02 + 0.24x^1 + 0.20x^2 + 0.00x^3 + 0.03x^4 + 0.00x^5 - 0.00x^6$

$P(0.55) = 0.1711$

c.  $x = 0.85$

Kalkulator Matriks Teman Tuan Mike

Menu utama

## Interpolasi Polinomial

**Input titik:**

Jumlah titik:

0.1 0.003
0.3 0.067
0.5 0.148
0.7 0.248
0.9 0.370
1.1 0.518
1.3 0.697

**Input variabel bebas:**

x:

**Operasi:**

**Hasil:**

Polinomial:  
 $P(x) = -0.02 + 0.24x^1 + 0.20x^2 + 0.00x^3 + 0.03x^4 + 0.00x^5 - 0.00x^6$

$P(0.85) = 0.3372$

d.  $x = 1.28$

Kalkulator Matriks Teman Tuan Mike

Menu utama

## Interpolasi Polinomial

**Input titik:**

Jumlah titik:

0.1 0.003
0.3 0.067
0.5 0.148
0.7 0.248
0.9 0.370
1.1 0.518
1.3 0.697

**Input variabel bebas:**

x:

**Operasi:**

**Hasil:**

Polinomial:  
 $P(x) = -0.02 + 0.24x^1 + 0.20x^2 + 0.00x^3 + 0.03x^4 + 0.00x^5 - 0.00x^6$

$P(1.28) = 0.6775$

### 4.2.2. Kasus 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

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

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

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 interpolasi polinomial untuk melakukan prediksi jumlah kasus baru Covid-19 pada tanggal-tanggal berikut:

- a. 16/07/2022

Kalkulator Matriks Teman Tuan Mike

Menu utama

## Interpolasi Polinomial

**Input titik:**

Jumlah titik:

6.567 12624
7 21807
7.258 38391
7.451 54517
7.548 51952
7.839 28228
8.161 35764
8.484 20813
8.709 12408
9 10534

**Input variabel bebas:**

x:

**Buttons:**

- Hitung
- Choose file
- Export result

**Interpolasi**

Polinomial:

$$P(x) = 7187066071658.64 - 9346993079172.96x^1 + 5334203055240.28x^2 - 1756810186361.37x^3 + 368550807175.53x^4 - 51131876760.13x^5 + 4695806315.43x^6 - 275474539.42x^7 + 9372849.24x^8 - 140993.71x^9$$

$P(7.53333333333333) = 52758.5723$

b. 10/08/2022

Kalkulator Matriks Teman Tuan Mike

Menu utama

## Interpolasi Polinomial

**Input titik:**

Jumlah titik:

6.567 12624
7 21807
7.258 38391
7.451 54517
7.548 51952
7.839 28228
8.161 35764
8.484 20813
8.709 12408
9 10534

**Input variabel bebas:**

x:

**Buttons:**

- Hitung
- Choose file
- Export result

**Interpolasi**

Polinomial:

$$P(x) = 7187066071658.64 - 9346993079172.96x^1 + 5334203055240.28x^2 - 1756810186361.37x^3 + 368550807175.53x^4 - 51131876760.13x^5 + 4695806315.43x^6 - 275474539.42x^7 + 9372849.24x^8 - 140993.71x^9$$

$P(8.33333333333334) = 35746.7500$

c. 05/09/2022

Kalkulator Matriks Teman Tuan Mike

Menu utama

## Interpolasi Polinomial

**Input titik:**

Jumlah titik:

6.567 12624
7 21807
7.258 38391
7.451 54517
7.548 51952
7.839 28228
8.161 35764
8.484 20813
8.709 12408
9 10534

**Input variabel bebas:**

x:

\*Nilai x diluar range titik interpolasi [6.57, 9.00]

Polinomial:

$$P(x) = 7187066071658.64 - 9346993079172.96x^1 + 5334203055240.28x^2 - 1756810186361.37x^3 + 368550807175.53x^4 - 51131876760.13x^5 + 4695806315.43x^6 - 275474539.42x^7 + 9372849.24x^8 - 140993.71x^9$$

- d. Masukan user lainnya berupa tanggal (desimal) yang sudah diolah dengan asumsi prediksi selalu dilakukan untuk tahun 2022.

Kalkulator Matriks Teman Tuan Mike

Menu utama

## Interpolasi Polinomial

**Input titik:**

Jumlah titik:

6.567 12624
7 21807
7.258 38391
7.451 54517
7.548 51952
7.839 28228
8.161 35764
8.484 20813
8.709 12408
9 10534

**Input variabel bebas:**

x:

\*Nilai x diluar range titik interpolasi [6.57, 9.00]

Polinomial:

$$P(x) = 7187066071658.64 - 9346993079172.96x^1 + 5334203055240.28x^2 - 1756810186361.37x^3 + 368550807175.53x^4 - 51131876760.13x^5 + 4695806315.43x^6 - 275474539.42x^7 + 9372849.24x^8 - 140993.71x^9$$

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

Kalkulator Matriks Teman Tuan Mike

Menu utama

### Interpolasi Polinomial

**Input titik:**

Jumlah titik: 6

0.0 0.0
0.4 0.418884230141255
0.8 0.5071579685304316
1.2 0.5609246748146804
1.6 0.5836856612868684
2.0 0.576651529751722

**Input variabel bebas:**

x: 1.532

Hitung

Choose file

Export result

Interpolasi

Polinomial:  
 $P(x) = 2.04x^1 - 3.55x^2 + 3.24x^3 - 1.42x^4 + 0.24x^5$   
 $P(1.532) = 0.5840$

### 4.3. Regresi 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.

Dari data-data tersebut, apabila diterapkan *Normal Estimation Equation for Multiple Linear Regression*, maka diperoleh sistem persamaan linear sebagai berikut.

$$\begin{aligned}
 20b_0 + 863.1b_1 + 1530.4b_2 + 587.84b_3 &= 19.42 \\
 863.1b_0 + 54876.89b_1 + 67000.09b_2 + 25283.395b_3 &= \\
 779.477 & \\
 1530.4b_0 + 67000.09b_1 + 117912.32b_2 + 44976.867b_3 &= 1483.437 \\
 587.84b_0 + 25283.395b_1 + 44976.867b_2 + 17278.5086b_3 &= \\
 571.1219 &
 \end{aligned}$$

Silahkan terapkan model-model ini pada *Multiple Quadratic Equation* juga dan bandingkan hasilnya. Sistem persamaan linear tidak akan diberikan untuk kasus ini.

### 4.3.1. Kasus 6a

Kalkulator Matriks Teman Tuan Mike

Menu utama

## Regresi Berganda

**Input sampel:**

Jumlah peubah (n): 3  
Jumlah sampel (m): 20

X:	Y:
23.2 76.8 29.38	0.94
47.4 86.6 29.35	1.10
31.5 76.9 29.63	1.10
10.6 86.3 29.56	1.10
11.2 86.0 29.48	0.91
73.3 76.3 29.40	0.87
75.4 77.9 29.28	0.78
96.6 78.7 29.29	0.82
107.4 86.8 29.03	0.95
54.9 70.9 29.37	

**Jenis regresi:**

Linier  
 Kuadratik

**Input variabel bebas:**

X<sub>1</sub>, X<sub>2</sub>, ..., X<sub>k</sub>:  
50 76 29.3

Taksir  
Choose file  
Export result

Regresi

Persamaan regresi linier:  

$$Y = -3.5078 - 0.0026X_1 + 0.0008X_2 + 0.1542X_3$$

Hasil taksiran regresi linier:  
 Pada titik X<sub>1</sub> = 50.00, X<sub>2</sub> = 76.00, X<sub>3</sub> = 29.30:  

$$Y = 0.9384$$

Kalkulator Matriks Teman Tuan Mike

Menu utama

## Regresi Berganda

**Input sampel:**

Jumlah peubah (n): 3  
Jumlah sampel (m): 20

X:	Y:
23.2 76.8 29.38	0.94
47.4 86.6 29.35	1.10
31.5 76.9 29.63	1.10
10.6 86.3 29.56	1.10
11.2 86.0 29.48	0.91
73.3 76.3 29.40	0.87
75.4 77.9 29.28	0.78
96.6 78.7 29.29	0.82
107.4 86.8 29.03	0.95
54.9 70.9 29.37	

**Jenis regresi:**

Linier  
 Kuadratik

**Input variabel bebas:**

X<sub>1</sub>, X<sub>2</sub>, ..., X<sub>k</sub>:  
50 76 29.3

Taksir  
Choose file  
Export result

Regresi

Persamaan regresi kuadratik:  

$$Y = -1146.4372 + 0.1840X_1 + 0.8386X_2 + 75.4503X_3 - 0.0000X_1^2 - 0.0002X_2^2 - 1.2404X_3^2 + 0.0000X_1X_2 - 0.0064X_1X_3 - 0.0272X_2X_3$$

Hasil taksiran regresi kuadratik:  
 Pada titik X<sub>1</sub> = 50.00, X<sub>2</sub> = 76.00, X<sub>3</sub> = 29.30:  

$$Y = 0.9439$$

### 4.3.2. Kasus 6b

Kalkulator Matriks Teman Tuan Mike

Menu utama

## Sistem Persamaan Linier

**Input matriks:**

Baris: 4    Kolom: 5

20	863.1	1530.4	587.84	19.42
863.1	54876.89	67000.09	25283.395	779.47
1530.4	67000.09	117912.32	44976.867	148
587.84	25283.395	44976.867	17278.5086	5

**Metode:**

- Eliminasi Gauss
- Eliminasi Gauss-Jordan
- Matriks Balikan
- Kaidah Cramer

**Solve**

$x_1 = -3.51$   
 $x_2 = -0.00$   
 $x_3 = 0.00$   
 $x_4 = 0.15$

**Choose file**  
**Export result**

### 4.4. Interpolasi *Bicubic Spline*

Diberikan matriks input dengan bentuk sebagai berikut. Format matriks masukan bukan mewakili nilai matriks, tetapi mengikuti format masukan pada bagian "Spesifikasi Tugas" nomor 7.

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

Tentukan nilai:

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

#### 4.4.1. Kasus 7a

Kalkulator Matriks Teman Tuan Mike

**Bicubic Spline Interpolation**

**Input matriks konfigurasi:**

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

**Format masukan matriks konfigurasi:**

f(0, 0)	f(1, 0)	f(0, 1)	f(1, 1)
fx(0, 0)	fx(1, 0)	fx(0, 1)	fx(1, 1)
fy(1, 0)	fy(0, 1)	fy(1, 1)	
fxy(0, 0)	fxy(1, 0)	fxy(0, 1)	fxy(1, 1)

**Input variabel bebas:**

x: 0 y: 0

**Buttons:**

- Choose file
- Export result
- Interpolasi

f(0.000000, 0.000000) = 21.000000

#### 4.4.2. Kasus 7b

Kalkulator Matriks Teman Tuan Mike

**Bicubic Spline Interpolation**

**Input matriks konfigurasi:**

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

**Format masukan matriks konfigurasi:**

f(0, 0)	f(1, 0)	f(0, 1)	f(1, 1)
fx(0, 0)	fx(1, 0)	fx(0, 1)	fx(1, 1)
fy(0, 0)	fy(1, 0)	fy(0, 1)	fy(1, 1)
fxy(0, 0)	fxy(1, 0)	fxy(0, 1)	fxy(1, 1)

**Input variabel bebas:**

x: 0.5 y: 0.5

**Buttons:**

- Choose file
- Export result
- Interpolasi

f(0.500000, 0.500000) = 87.796875

#### 4.4.3. Kasus 7c

Kalkulator Matriks Teman Tuan Mike

Menu utama

### Bicubic Spline Interpolation

**Input matriks konfigurasi:**

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

Format masukan matriks konfigurasi:

f(0, 0)	f(1, 0)	f(0, 1)	f(1, 1)
fx(0, 0)	fx(1, 0)	fx(0, 1)	fx(1, 1)
fy(1, 0)	fy(0, 1)	fy(1, 1)	
fxy(0, 0)	fxy(1, 0)	fxy(0, 1)	fxy(1, 1)

**Input variabel bebas:**

x: 0.25    y: 0.75

Choose file    Export result

**Interpolasi**

$f(0,250000, 0,750000) = 117,732178$

#### 4.4.4. Kasus 7d

Kalkulator Matriks Teman Tuan Mike

Menu utama

### Bicubic Spline Interpolation

**Input matriks konfigurasi:**

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

Format masukan matriks konfigurasi:

f(0, 0)	f(1, 0)	f(0, 1)	f(1, 1)
fx(0, 0)	fx(1, 0)	fx(0, 1)	fx(1, 1)
fy(0, 0)	fy(1, 0)	fy(0, 1)	fy(1, 1)
fxy(0, 0)	fxy(1, 0)	fxy(0, 1)	fxy(1, 1)

**Input variabel bebas:**

x: 0.1    y: 0.9

Choose file    Export result

**Interpolasi**

$f(0,100000, 0,900000) = 128,575187$

## BAB 5

# KESIMPULAN, SARAN, KOMENTAR, DAN REFLEKSI

### A. Kesimpulan

Berbagai permasalahan matematis seringkali ditemukan di dunia nyata. Oleh sebab itu, diperlukan suatu metode yang efisien dan akurat yang mampu membantu proses perhitungan permasalahan matematis tersebut. Salah satu contoh permasalahan matematis yang banyak ditemukan dalam kehidupan sehari-hari adalah sistem persamaan linier. Terdapat banyak metode untuk menyelesaikan sistem persamaan linier, salah satunya dengan memanfaatkan matriks. Dalam memanfaatkan matriks, terdapat 4 metode yang dapat digunakan untuk menemukan suatu solusi dari sistem persamaan linier, yakni metode Eliminasi Gauss, Eliminasi Gauss-Jordan, Balikan, dan Cramer. Dalam pemrosesan metode-metode tersebut, perlu didefinisikan terlebih dahulu fungsi-fungsi primitif untuk menjadi pondasi dalam membangun metode yang bersangkutan. Contoh fungsi-fungsi primitifnya seperti perhitungan determinan, operasi OBE, dan sebagainya. Sebagai mahasiswa yang mengambil bidang studi keinformatikaan, tim peneliti mampu menciptakan sebuah program komputer yang dapat memanipulasi SPL dalam bentuk matriks untuk menghasilkan solusi dalam bahasa pemrograman Java.

Ada 3 kemungkinan solusi dalam suatu sistem persamaan linier, yakni banyak solusi (hasilnya dalam bentuk parametrik), tidak ada solusi (tidak ada nilai yang memenuhi sistem persamaan linier tersebut), serta memiliki satu solusi unik. Menganalisis keempat metode yang dapat digunakan, diketahui bahwa dalam beberapa kasus, metode Balikan dan Cramer tidak dapat digunakan untuk menemukan solusi, yakni ketika determinan suatu matriks persegi yang hendak diselesaikan bukan nol. Untuk mengatasinya, harus digunakan metode Eliminasi Gauss maupun Eliminasi Gauss-Jordan sehingga didapatkan solusi berupa persamaan parametrik. Pengaplikasian sistem persamaan linier juga dapat digunakan dalam pengaplikasian interpolasi polinom, *bicubic spline interpolation*, serta regresi linier dan kuadratik berganda. Interpolasi polinom digunakan untuk memprediksi suatu nilai berdasarkan nilai  $x$  dalam persamaan polinomial yang konstantanya dapat dicari menggunakan metode Eliminasi Gauss. *Bicubic spline interpolation* dapat digunakan untuk memprediksi suatu nilai pada titik tertentu dengan mempertimbangkan 16 titik di sekitarnya.

Sedangkan regresi linier berganda dan regresi kuadratik berganda dapat digunakan untuk memprediksi nilai yang dipengaruhi oleh lebih dari satu variabel sekaligus.

Melalui tugas besar dari mata kuliah Aljabar Linier dan Geometri IF2123 ini, besar harapan kami bahwa program yang telah kami buat mampu membantu penggunaanya untuk menyelesaikan persamaan matematis baik dalam pembelajaran maupun pada dunia nyata secara efisien dan akurat.

## B. Saran

Setelah proses pengerjaan tugas besar Aljabar Linier dan Geometri IF2123, tim peneliti memiliki beberapa saran perbaikan yang diharapkan dapat diterapkan oleh para peneliti lain kedepannya:

- Memperbaiki penampilan GUI

Meningkatkan tampilan antarmuka pengguna (GUI) agar lebih menarik, intuitif, dan mudah digunakan. Beberapa hal yang dapat diperbaiki melibatkan penggunaan layout yang rapi maupun elemen visual seperti ikon dan animasi untuk meningkatkan estetika.

- Belajar OOP dan *modular programming* (*libraries* dan *build tools*) lebih awal

Menguasai konsep *Object-Oriented Programming* (OOP) dan *modular programming* lebih awal akan memudahkan pengembangan program tugas besar ini karena mampu membantu dalam pengorganisasian kode ke dalam objek-objek, sementara *modular programming* memungkinkan kode untuk dibagi menjadi bagian-bagian kecil yang mudah dikelola dan diuji. *Libraries* dan *build tools* seperti Maven juga membantu dalam pengelolaan GUI proyek.

- Nama fungsi tidak konsisten

Ketidakkonsistenan nama fungsi dapat membuat kode sulit dipahami sehingga sebaiknya digunakan pola penamaan yang jelas dan seragam agar mudah diikuti, misalnya menggunakan variabel dengan bahasa yang sepenuhnya dalam bahasa Inggris.

- Mengatur waktu dengan baik agar bisa mengerjakan soal bonus

Penting untuk manajemen waktu agar tugas besar dapat selesai tepat waktu dan memiliki waktu lebih untuk mengerjakan soal bonus.

## C. Komentar

Chris: Jangan lagi kerjain 2 bonus H-1 😊

Grace: Terima kasih teman-teman semua tanpa kalian aku hanyalah butiran debu T\_T

Karol: System.out.println("dua tiga tutup botol, lov yu ol");

## D. Refleksi

Melalui tugas besar ini, tim peneliti dapat mengimplementasikan pemahaman yang didapat dari mata kuliah Aljabar Linier dan Geometri 2123, yakni mengenai materi sistem persamaan linier dan penyelesaiannya menggunakan berbagai metode matriks. Penggunaan bahasa pemrograman dan paradigma pemrograman yang belum pernah dipelajari sebelumnya (Java) meningkatkan tingkat kesulitan dalam penggerjaan tugas besar ini karena tim peneliti dituntut untuk belajar dan beradaptasi dengan cepat terhadap syntax Java. Selain kemampuan pemrograman, tim peneliti juga dituntut untuk mampu bekerjasama dan berkomunikasi dengan baik dalam kelompok sehingga tugas besar dapat diselesaikan dengan baik. Tim peneliti menyadari bahwa terdapat masih banyak kekurangan dalam tugas besar ini. Namun, besar harapan bagi tim peneliti untuk memperbaikinya pada kesempatan yang akan datang.

## DAFTAR PUSTAKA

- Gerald, C. F., & Wheatley, P. O. (2004). *Applied numerical analysis* (7th ed.). Pearson.  
<https://archive.org/details/applied-numerical-analysis-by-gerald--c.-f-and-wheatley-7th-edn>
- Rinaldi Munir. (2024). Sistem persamaan linier (Bagian 1: Metode Eliminasi Gauss). Institut Teknologi Bandung.  
<https://informatika.stei.itb.ac.id/~rinaldi.munir/AljabarGeometri/2023-2024/Algeo-03-Sistem-Persamaan-Linier-2023.pdf>
- Rinaldi Munir. (2024). Sistem persamaan linier (Bagian 2: Tiga kemungkinan solusi sistem persamaan linier). Institut Teknologi Bandung.  
<https://informatika.stei.itb.ac.id/~rinaldi.munir/AljabarGeometri/2023-2024/Algeo-04-Tiga-Kemungkinan-Solusi-SPL-2023.pdf>
- Rinaldi Munir. (2024). Sistem persamaan linier (Bagian 3: Metode Eliminasi Gauss-Jordan). Institut Teknologi Bandung.  
<https://informatika.stei.itb.ac.id/~rinaldi.munir/AljabarGeometri/2023-2024/Algeo-05-Sistem-Persamaan-Linier-2-2023.pdf>
- Rowe, D. P. (2018). *BiLinear, Bicubic, and In Between Spline Interpolation*. Marquette University.  
[mssc.mu.edu/~daniel/pubs/RoweTalkMSCS\\_BiCubic.pdf](http://mssc.mu.edu/~daniel/pubs/RoweTalkMSCS_BiCubic.pdf)
- Strang, G. (2016). *Introduction to linear algebra* (5th ed.). Wellesley-Cambridge Press.  
<https://math.mit.edu/~gs/linearalgebra/>

## TAUTAN REPOSITORY

<https://github.com/karolyangqian/Algeo01-23077>

## TAUTAN VIDEO

 Algeo-23077