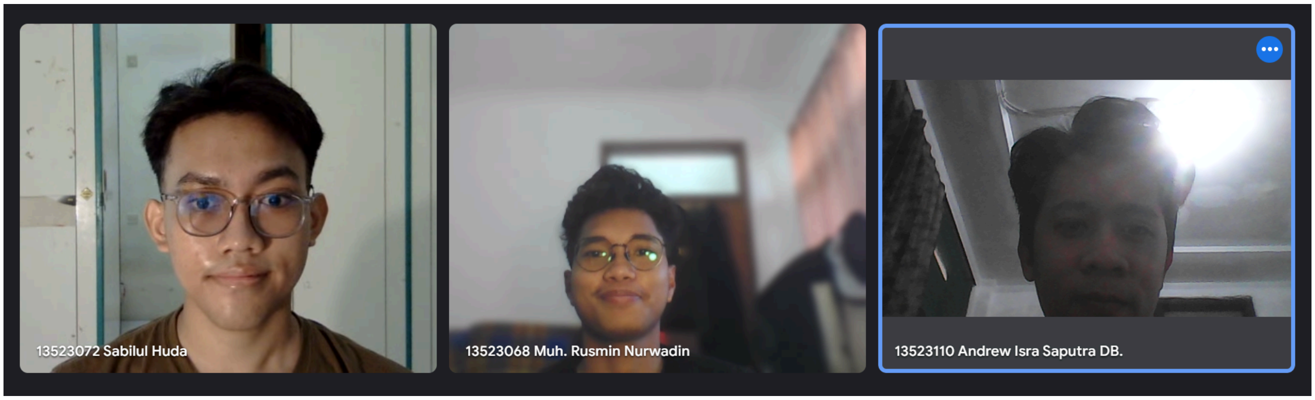
TUGAS BESAR 1 IF2123
Aljabar Linear dan Geometri



Diampu oleh:

Dr. Ir. Rinaldi Munir, M.T.

Disusun oleh:

Kelompok 20

Muh. Rusmin Nurwadin	13523068
Sabilul Huda	13523072
Andrew Isra Saputra DB	13523110

SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA
INSTITUT TEKNOLOGI BANDUNG
2024

BAB 1

DESKRIPSI MASALAH

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

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

BAB 2

TEORI SINGKAT

2.1. Sistem Persamaan Linier

2.1.1 Metode Eliminasi Gauss

Metode yang dinamai berdasarkan nama matematikawan Carl Friedrich Gauss (1777-1855) ini adalah salah satu algoritma yang digunakan untuk menyelesaikan sistem persamaan linier (SPL). Metode eliminasi Gauss bekerja dengan mengubah matriks augmented menjadi bentuk matriks eselon baris dengan menggunakan operasi baris elementer. Setelah matriks mencapai bentuk tersebut, teknik penyulihan mundur digunakan untuk menemukan nilai variabel x.

Metode eliminasi Gauss adalah langkah penting dalam aljabar linear, terutama dalam menyelesaikan sistem persamaan linier. Prosesnya dimulai dengan menyusun matriks augmented, yang menggabungkan koefisien dari variabel dan konstanta dari persamaan. Melalui operasi baris elementer, seperti pertukaran baris, pengalian baris dengan skalar, dan penjumlahan baris, matriks tersebut diubah menjadi bentuk eselon baris.

Setelah matriks dalam bentuk eselon baris, langkah selanjutnya adalah teknik penyulihan mundur, di mana nilai variabel ditentukan dari persamaan yang telah dihasilkan. Metode ini efisien dan sering digunakan dalam berbagai aplikasi matematika dan teknik, karena dapat menangani sistem persamaan dengan sejumlah besar variabel dan persamaan.

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

2.1.2. Metode Eliminasi Gauss Jordan

Metode ini adalah variasi dari eliminasi Gauss yang diperkenalkan oleh Wilhelm Jordan pada tahun 1888. Metode eliminasi Gauss-Jordan mengubah matriks augmented menjadi bentuk matriks eselon baris tereduksi, di mana elemen-elemen di atas dan di bawah elemen utama (yang bernilai 1) menjadi nol. Dengan cara ini, tidak perlu lagi melakukan teknik penyulihan mundur untuk menemukan nilai variabel x.

Eliminasi Gauss-Jordan adalah metode yang lebih lanjut dari eliminasi Gauss, yang bertujuan untuk menyederhanakan penyelesaian sistem persamaan linier. Dalam proses ini, matriks augmented diubah sehingga semua elemen di atas dan di bawah elemen utama pada setiap kolom menjadi nol. Hal ini membuat sistem persamaan menjadi lebih mudah dibaca, dan nilai variabel dapat ditentukan langsung dari baris terakhir matriks.

Metode ini sangat berguna dalam aplikasi yang memerlukan penyelesaian sistem persamaan secara langsung tanpa memerlukan langkah tambahan untuk menyusun kembali solusi, sehingga mempercepat proses dan mengurangi potensi kesalahan. Eliminasi Gauss-Jordan juga digunakan dalam berbagai bidang, seperti komputer sains, statistik, dan teknik, karena kemampuannya untuk menangani sistem yang lebih kompleks secara efisien.

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

2.1.3. Metode Matriks Balikan

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

2.1.4. Kaidah Cramer

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

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

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

2.2. Determinan

Determinant adalah suatu nilai skalar yang dapat dihitung dari elemen-elemen sebuah matriks persegi (matriks yang memiliki jumlah baris dan kolom yang sama). Determinan memberikan informasi penting tentang matriks tersebut, seperti apakah matriks tersebut invertible (dapat dibalik) atau tidak. Jika determinan sebuah matriks tidak sama dengan nol, maka matriks tersebut memiliki invers, dan sistem persamaan linier yang diwakili oleh matriks tersebut memiliki solusi unik. Selain itu, determinan juga terkait dengan volume dan geometri, seperti bagaimana matriks tersebut mempengaruhi ruang vektor.

2.2.1. Metode Reduksi Baris

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

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

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

Sehingga didapatkan

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

dimana p menyatakan banyaknya operasi pertukaran baris di dalam OBE.

2.2.2. Metode Ekspansi Kofaktor

Didefinisikan sebuah matriks persegi A sebagai berikut

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

Definisikan notasi Mij sebagai minor dari entri aij, yaitu determinan dari sub-matriks yang elemen-elemennya adalah elemen matriks A yang tidak berada pada baris i dan kolom j.
Lalu didefinisikan juga Cij sebagai kofaktor dari entri aij, yaitu

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

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

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

Secara baris

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

Secara kolom

2.3. Balikan Matriks

Invers dari sebuah matriks adalah matriks yang, ketika dikalikan dengan matriks asalnya, menghasilkan matriks identitas. Agar sebuah matriks memiliki invers, ada dua syarat yang harus dipenuhi: determinan matriks tersebut tidak boleh nol, dan matriks harus berbentuk persegi. Ada beberapa metode untuk mencari invers matriks, di antaranya adalah menggunakan matriks adjoin dan metode eliminasi Gauss-Jordan.

2.3.1. Metode Matriks Adjoin

Didefinisikan suatu matriks sebagai berikut

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

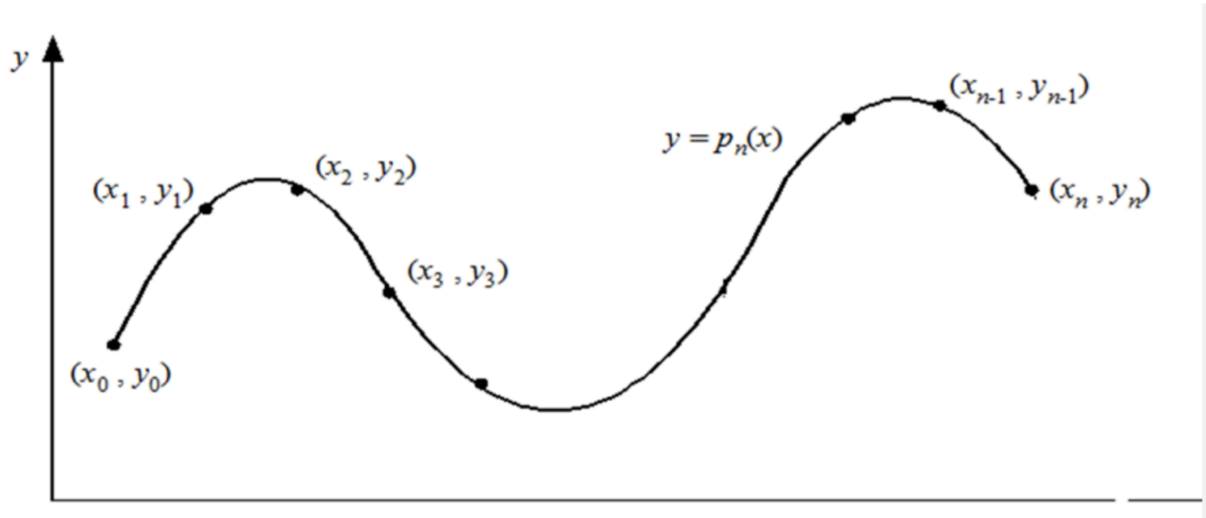
Dengan demikian, entri kofaktor pada baris i dan kolom j dapat dihitung dengan rumus (-1)^(i+j) dikali dengan determinan dari matriks minor yang diperoleh dengan menghilangkan elemen pada baris i dan kolom j dari matriks asal. Setelah kofaktor suatu matriks dihitung, invers matriks dapat diperoleh dengan mentranspose matriks adjoin dan mengalikan setiap elemen matriks tersebut dengan determinan dari matriks asal.

2.3.2. Metode Eliminasi Gauss Jordan

Diberikan suatu matriks \(A \), invers dari matriks tersebut dapat ditemukan dengan menerapkan metode eliminasi Gauss-Jordan pada matriks \(A \) yang diaugmented dengan matriks identitas. Proses ini akan menghasilkan matriks identitas di sisi \(A \) dan invers matriks \(A \) di sisi matriks identitas.

2.4. Interpolasi Polinom

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



2.5. Bicubic Spline Interpolation

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

Semisal untuk mencari nilai (x,y) yang berada pada rentang (0,0), (0,1), (1,0), hingga (1,1); kita akan memerlukan 4 titik utama yaitu f(0, 0), f(1, 0), f(0, 1), f(1, 1), serta 12 titik turunan berarah pada 4 titik utama tersebut, dalam hal ini 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).

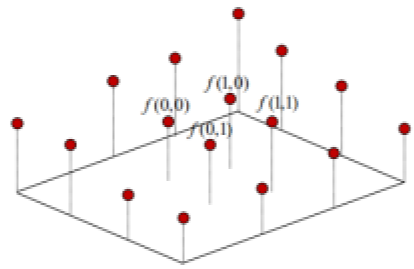
Normalization: $f(0,0), f(1,0)$

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

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

$x = -1, 0, 1, 2$

Solve: a_{ij}



$$y = Xa$$

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

Dari ekspansi Sum tersebut diperoleh matriks X seperti berikut. Lalu akan dicari matriks *alpha* menggunakan persamaan tersebut. Lalu terakhir, akan dikalikan matriks alpha tersebut dengan polinom x dan y sehingga mendapatkan nilai dari interpolasi di titik (x,y).

$$p(x,y) = \begin{bmatrix} 1 & x & x^2 & x^3 \end{bmatrix} \begin{bmatrix} a_{00} & a_{01} & a_{02} & a_{03} \\ a_{10} & a_{11} & a_{12} & a_{13} \\ a_{20} & a_{21} & a_{22} & a_{23} \\ a_{30} & a_{31} & a_{32} & a_{33} \end{bmatrix} \begin{bmatrix} 1 \\ y \\ y^2 \\ y^3 \end{bmatrix}.$$

2.6. Regresi

2.6.1 Regresi Linier Berganda

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

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

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

2.6.2 Regresi Kuadratik Bergand

Regresi kuadratik berganda adalah suatu metode analisis statistik yang digunakan untuk memodelkan hubungan antara satu variabel dependen (target) dan dua atau lebih variabel independen (prediktor) dengan menggunakan persamaan kuadratik. Dalam konteks ini, "kuadratik" merujuk pada adanya suku-suku yang melibatkan pangkat dua dari variabel independen.

Persamaan Umum :

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_1^2 + \beta_4 x_2^2 + \beta_5 x_1 x_2 + \ldots + \beta_n x_n + \epsilon$$

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

2.7. Image Processing

Image processing ini melibatkan algoritma Bicubic Spline Interpolation (saat ini interpolasi yang paling halus merupakan algoritma Bicubic Spline Interpolation). Dimana nilai dari matriks X tadi dikalikan dengan matriks D yang menghasilkan matriks berikut. Adapun perbedaan dari Bicubic Spline Interpolation biasanya, yaitu pada pengali matriksnya yang merupakan nilai RGB di sekitar titik interpolasi. Hal ini karena keterbatasan untuk menghitung turunan berarah dari sebuah image.

$$a = X^{-1}DI$$

$$\begin{bmatrix} a_{00} \\ a_{10} \\ a_{20} \\ a_{30} \\ a_{01} \\ a_{11} \\ a_{21} \\ a_{31} \\ a_{02} \\ a_{12} \\ a_{22} \\ a_{32} \\ a_{03} \\ a_{13} \\ a_{23} \\ a_{33} \end{bmatrix} = \frac{1}{16}\begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 16 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & -8 & 0 & 8 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 16 & -40 & 32 & -8 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & -8 & 24 & -24 & 8 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & -8 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 8 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & -4 & 0 & 0 & -4 & 4 & 0 & 0 & 0 & 0 & 4 & 0 & 0 & 0 & 0 & 0 \\
0 & 32 & -20 & 0 & 8 & -4 & -4 & 0 & 0 & -24 & 16 & -4 & 0 & 0 & 0 & 0 \\
0 & -20 & 12 & 0 & -4 & 0 & 4 & 0 & 0 & 16 & -12 & 4 & 0 & 0 & 0 & 0 \\
0 & 16 & 0 & 0 & 0 & -40 & 0 & 0 & 0 & 32 & 0 & 0 & 0 & -8 & 0 & 0 \\
0 & 8 & 0 & 0 & 32 & -4 & -24 & 0 & -20 & -4 & 0 & 0 & 0 & 0 & -4 & 0 \\
0 & -64 & 40 & 0 & -64 & 96 & -68 & 24 & 40 & -68 & -16 & -16 & 0 & 24 & -16 & 0 \\
0 & 40 & -24 & 0 & 32 & -52 & 52 & -24 & -20 & 40 & 16 & 16 & 0 & -16 & 12 & 0 \\
0 & -8 & 0 & 0 & 0 & 24 & 0 & 0 & 0 & -24 & 0 & 0 & 0 & 8 & 0 & 4 \\
0 & -4 & 0 & 0 & -20 & 0 & 16 & 0 & 12 & 4 & -12 & 0 & 0 & 0 & 4 & -4 \\
0 & 32 & -20 & 0 & 40 & -52 & 40 & -16 & -24 & 52 & -52 & 12 & 0 & -24 & 16 & -4 \\
0 & -20 & 12 & 0 & -20 & 28 & -32 & 16 & 12 & -32 & 40 & -12 & 0 & 16 & -12 & 4
\end{bmatrix}\begin{bmatrix} I(-1,-1) \\ I(0,-1) \\ I(1,-1) \\ I(2,-1) \\ I(-1,0) \\ I(0,0) \\ I(1,0) \\ I(2,0) \\ I(-1,1) \\ I(0,1) \\ I(1,1) \\ I(2,1) \\ I(-1,2) \\ I(0,2) \\ I(1,2) \\ I(2,2) \end{bmatrix}$$

recycle matrix

BAB 3

IMPLEMENTASI PUSTAKA DAN PROGRAM DALAM JAVA

Matrix.java

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

- **Atribut**

Atribut	Deskripsi
`double[][] matrix`	Elemen dari matriks yang digunakan
`int rowNum`	Jumlah baris yang digunakan pada matriks berupa integer
`int colNum`	Jumlah kolom yang digunakan pada matriks berupa integer
`double [] SPLsolution`	Array untuk menyimpan solusi SPL
`string[] infiniteSol`	Array yang menyipan solusi parametrik (variabel)
`boolean infiniteSol`	Penanda apakah solusi infinite atau tidak
`double[][] infiniteKoef`	Array untuk menyimpan koefisien solusi parametrik
`StringBuilder output`	Array string yang menyimpan semua *output,* agar dapat disimpan pada file

- **Konstruktor**

Konstruktor	Deskripsi
`public Matrix(int rowNum, int colNum)`	Dibuat matriks dengan mengisi atribut rowNum = rowNum dan colNum = colNum, dan menginisiasi array matrix berukuran rowNum x colNum.
`public Matrix()`	Membuat objek matriks kosong dengan rowNum dan colNum diset menjadi 0.

- **Method**

Method	Deskripsi

`public void readMatrix(String type)`	Membaca input matriks secara manual atau dari file .txt, dan mengisi atribut matrix sesuai input.
`public void writeMatrix()`	Menampilkan isi matriks pada layar.
`public double[][] addMatrix(Matrix other)`	Menambahkan dua matriks dengan mengembalikan matriks hasil penjumlahan dari `this.matrix` dan `other.matrix`.
`public double[][] substractMatrix(Matrix other)`	Mengurangi dua matriks dengan mengembalikan matriks hasil pengurangan dari `this.matrix` dan `other.matrix`.
`public double[][] multiplyMatrix(Matrix other)`	Mengalikan matriks `this.matrix` dengan matriks `other.matrix`. Menghasilkan kesalahan jika jumlah kolom matriks pertama tidak sama dengan jumlah baris matriks kedua.
`public double[][] multiplyConstant(double constant)`	Mengalikan setiap elemen dalam matriks dengan sebuah konstanta, dan mengembalikan hasilnya dalam bentuk matriks baru.
`public double[][] transposeMatrix()`	Menghasilkan matriks transpose dari matriks asli. Matriks transpose diperoleh dengan menukar baris dengan kolom pada matriks.
`public Matrix getMinor(int row, int col)`	Menghasilkan matriks minor, yaitu matriks yang diperoleh dengan menghapus satu baris dan satu kolom tertentu dari matriks asli.
`public double DeterminantUsingCofactor()`	Menghitung determinan matriks menggunakan metode kofaktor. Matriks harus berbentuk persegi (jumlah baris = jumlah kolom). Menggunakan rekursi untuk matriks berukuran lebih dari 2x2.
`public double DeterminantUsingRowReduction()`	Menghitung determinan menggunakan metode eliminasi baris atau Operasi Baris Elementer (OBE). Membentuk matriks segitiga atas dan menghitung hasil dari perkalian elemen diagonal.
`public double[][] InverseUsingAdjoin()`	Menghasilkan invers dari matriks menggunakan metode adjoin. Matriks kofaktor dihitung, kemudian di-transpose untuk mendapatkan adjoint, dan akhirnya dikalikan dengan kebalikan dari determinan matriks. Invers tidak ada jika determinan 0.
`public double[][] InverseUsingGaussJordan()`	Menghasilkan invers dari matriks menggunakan metode Gauss-Jordan. Matriks diperluas (augmented) dengan matriks identitas, dan melalui serangkaian operasi baris, matriks diubah menjadi bentuk yang menghasilkan invers.

`solveSPLGaussMethod()`	Menyelesaikan Sistem Persamaan Linear (SPL) menggunakan metode eliminasi Gauss, mencari solusi SPL dengan eliminasi dan back substitution.
`solveSPLGaussJordanMethod()`	Menyelesaikan SPL menggunakan metode eliminasi Gauss-Jordan, menghasilkan solusi dengan mengubah matriks menjadi bentuk baris eselon tereduksi.
`solveSPLInversMatrix()`	Menyelesaikan SPL dengan metode invers matriks, menggunakan matriks koefisien A dan matriks konstanta B untuk menghitung solusi SPL.
`solveSPLUsingCramer()`	Menyelesaikan SPL menggunakan aturan Cramer, menghitung determinan matriks koefisien A dan matriks Ai untuk mendapatkan solusi.
`public void solveManySolution()`	Menyelesaikan sistem persamaan linear yang memiliki banyak solusi (solusi tak hingga). Metode ini menghitung koefisien variabel bebas dan memberikan solusi parametris dalam bentuk simbolik.
`printSol()`	Menampilkan solusi SPL pada layar. Jika solusi tak hingga, akan menampilkan solusi parametrik. Jika solusi tunggal, akan menampilkan nilai masing-masing variabel.

Interpolasi.java

Class ini digunakan untuk menyelesaikan persoalan interpolasi polinom.

- **Atribut**

Atribut	Deskripsi
`private int n`	Menyimpan jumlah titik yang digunakan dalam interpolasi.
`private Matrix point`	Objek `Matrix` yang menyimpan koordinat titik (x, y) untuk interpolasi.
`private Matrix mat`	Objek `Matrix` yang merepresentasikan matriks augmented dari persamaan interpolasi.
`private double x`	Nilai x yang ingin dicari taksiran nilai y-nya.
`private double[] solOfInterpolation`	Array yang menyimpan solusi dari persamaan interpolasi.
`static Scanner input`	Objek `Scanner` untuk membaca input dari pengguna.
`StringBuilder output`	Objek `StringBuilder` untuk menyimpan output hasil

	interpolasi dalam format string.

- **Konstruktor**

Kontstruktor	Deskripsi
`public Interpolasi()`	Konstruktor yang menginisialisasi atribut `n` menjadi 0 dan `output` sebagai `StringBuilder` baru.

- **Method**

Method	Deskripsi
`public void readInterpolasi()`	Memungkinkan pengguna untuk memasukkan titik-titik interpolasi baik melalui keyboard maupun dari file. Mengisi matriks titik (`point`) dan matriks persamaan (`mat`).
`public void solveInterpolasi()`	Menghitung solusi dari persamaan interpolasi menggunakan metode Gauss-Jordan pada matriks (`mat`). Menyimpan solusi dalam array `solOfInterpolation`.
`public void printSol()`	Mencetak persamaan interpolasi berdasarkan solusi yang dihitung, menampilkan koefisien dalam format yang sesuai.
`public void printfc()`	Menghitung dan mencetak nilai taksiran `y` untuk nilai `x` yang telah dimasukkan oleh pengguna, berdasarkan solusi dari persamaan interpolasi.

Regresi.java

Class ini digunakan untuk menyelesaikan persoalan regresi linear berganda dan regresi kuadratik berganda.

- **Atribut**

Atribut	Deskripsi
`static Scanner scanner`	Scanner untuk membaca input dari pengguna.
`static int n`	Jumlah baris (data) yang diinputkan.
`static int m`	Jumlah kolom (variabel x) yang diinputkan.

`static Matrix matrix_X`	Matriks untuk menyimpan data variabel x.
`static Matrix matrix_Y`	Matriks untuk menyimpan data variabel y (target).
`static Matrix Data`	Matriks untuk menyimpan seluruh inputan data.
`static double[] Targer`	Array untuk menyimpan nilai target yang ingin dihampiri.
`public static StringBuilder output`	StringBuilder untuk menyimpan hasil output.

- **Konstruktor**

-

- **Method**

Method	Deskripsi
`public static readMatrix`	Menu awal untuk membaca inputan dari pengguna
`public static void readMatrixManual`	Membaca input secara manual, langsug dari *keyboard*
`public static readMatrixFIle`	Membaca input dari file .txt
`public static void setElmt_XY`	Membuat matriks x dan matriks y
`public static MultipleLinearRegression`	Menyelesaikan persoalan regresi linier berganda
`public static MultipleQuadraticRegression`	Menyelesaikan persoalan regresi kuadratik berganda

BicubicSplineInterpolation.java

Class ini digunakan untuk menyelesaikan persoalan bicubic interpolation.

- **Atribut**

Atribut	Deskripsi
`static Scanner scanner`	Objek untuk membaca input dari pengguna
`public static StringBuilder`	StringBuilder untuk menyimpan hasil output.

`output`	

- **Konstruktor**

-

- **Method**

Method	Deskripsi
`public static Object[] readManualInput()`	Membaca input dari pengguna melalui terminal untuk dimasukkan sebagai matrix dan titik interpolasi
`public static Object[] readFromFile(String filename) throws IOException`	Membaca file.txt yang akan diubah menjadi matriks dan titik interpolasi
`public static double bicubicInterpolation(Matrix coeffs, double a, double b)`	Menghitung nilai interpolasi di suatu titik dengan rentang 0<= x,y <=1
`public static Matrix getCoefficients(double[] values, Matrix A_inv)`	Menghitung koefisien alpha untuk digunakan pada interpolasi

ResizeImage.java

Class ini digunakan untuk memperbesar citra dengan bicubic interpolation.

- **Atribut**

Method	Deskripsi
`public ResizeImage(String imagePath)`	Membaca file image untuk mengambil panjang dan lebar pixelnya
`public static int[][][] readImage(String fileName)`	Membaca file image serta mengembalikan matriks 3 dimensi yang merepresentasikan nilai RGB pada image
`public static int bicubic(Matrix matrixRGBtemp, double a, double b)`	Menghitung nilai interpolasi setiap titik di image sesuai kebutuhan

- **Konstruktor**

-
- **Method**

Method	Deskripsi
`public static void ImageProcessingDriver()`	Prosedur untuk memperbesar citra dengan menggunakan bicubic interpolation atau double cubic interpolation. Input berupa masukan nama file pada folder test/image/, nama file hasil output image, faktor pembesaran lebar, dan faktor pembesaran tinggi. Output berupa file dengan nama yang telah diinput.
`public static void bicubic(String readDir, String writeDir, int heightFactor, int widthFactor)`	Prosedur untuk memperbesar citra dengan menggunakan bicubic interpolation.
`public static double bicubicInterpolation(int[] zValue, double a, double b)`	Fungsi untuk menghitung nilai hasil bicubic interpolation.
`public static int checkValue(double val)`	Fungsi untuk mengecek apakah nilai val melebihi 0xff atau kurang dari 0x0.
`public static void doubleCubic(String readDir, String writeDir, int heightFactor, int widthFactor)`	Prosedur untuk memperbesar citra dengan menggunakan cubic interpolation dua kali.
`public static int[][] interpolatePoints(int width, int height, int factor, boolean interpolateHeight)`	Fungsi untuk menghitung nilai elemen matriks hasil perbesaran interpolasi cubic.
`public static int getValue(int index, double x, boolean interpolateHeight)`	Fungsi untuk menghitung nilai ARGB hasil interpolasi.
`public static int RGBValue(int[] ordinate, double x)`	Fungsi untuk menghitung nilai pada titik f(x).

Utility.java

Class ini digunakan sebagai utility, umumnya digunakan untuk validasi input.

- **Atribut**

Method	Deskripsi
`static Scanner scan`	Menerima inputan pengguna

- **Konstruktor**

-

- **Method**

Method	Deskripsi
`public static boolean cek_int(String a)`	Mengecek apakah integer atau tidak
`public static int getValidChoice(int min, int max)`	Validasi inputan sesuai atau tidak
`public static void validasiFile(String output)`	Validasi sebelum menyipan output
`public static saveOutputToFile(String output)`	Menyimpan output pada file
`public static roundMatrixELements(Matrix M)`	Membulatkan suatu matriks
`public static void roundArrayElements(double[] array)`	Membulatkan suatu array
`public static double roundElmt`	Membulatkan elmen matriks

UI.java

Class ini digunakan untuk menampilkan beberapa bagian pada program.

- **Atribut**

-

- **Konstruktor**

-

- **Method**

Method	Deskripsi
`public static void Menu()`	Menampilkan pilihan menu
`public static void SPL_Menu()`	Menampilkan pilihan SPL
`public static void Det_Menu()`	Menampilkan pilihan determinan
`public static void inverse_Menu()`	Menampilkan pilihan inverse
`public static void reg_Menu()`	Menampilkan pilihan regresi
`public static void MenuArt()`	Menampilkan tulisan “MENU”

Main.java

Class ini digunakan sebagai penyatu semua program.

- **Atribut**

Atribut	Deskripsi
`static Scanner scanner`	Objek untuk membaca input dari pengguna

- **Konstruktor**

-

- **Method**

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

BAB 4

EKSPERIMEN

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

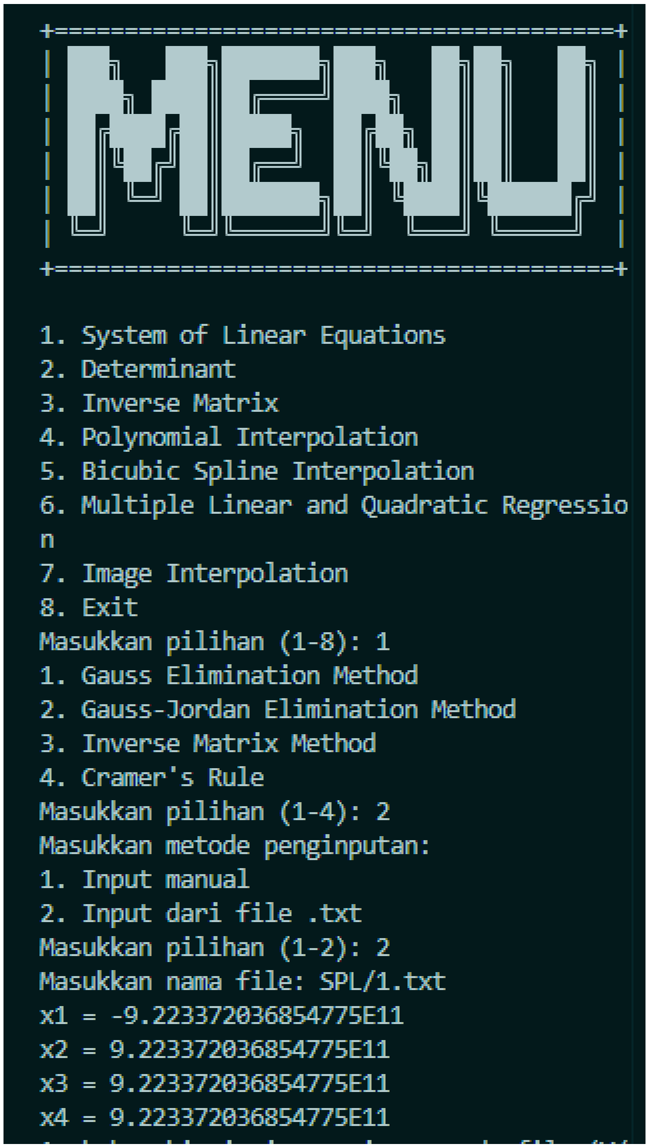
a. Test case 1

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

- Metode Gauss:

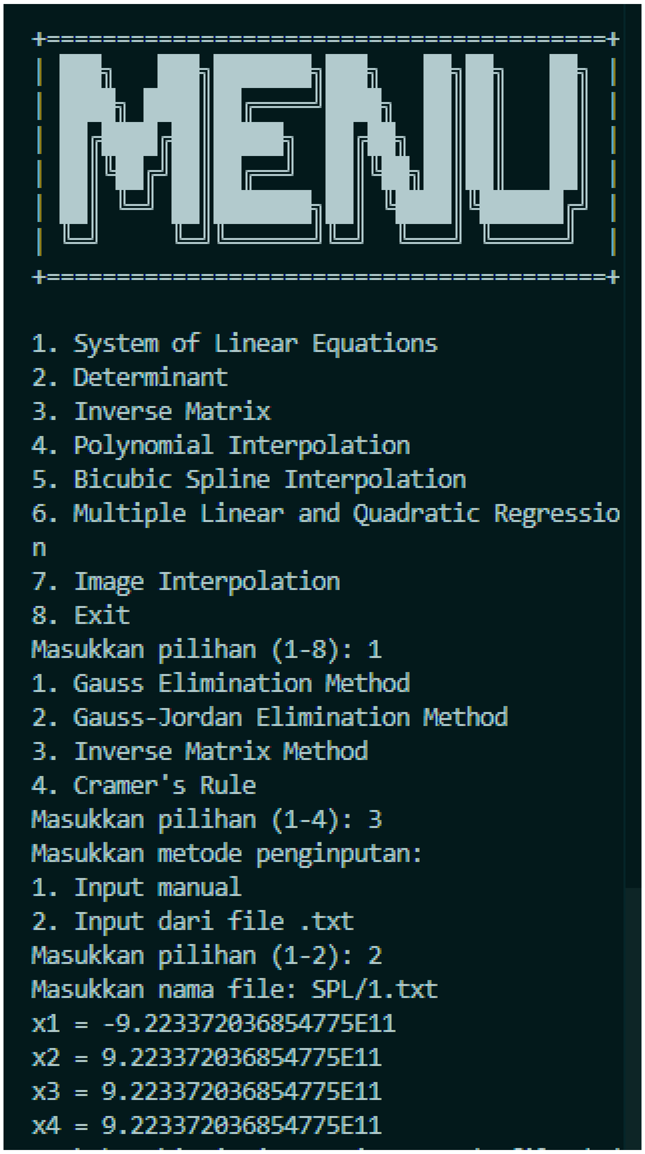
```
+=========================================+
+=========================================+

1. System of Linear Equations
2. Determinant
3. Inverse Matrix
4. Polynomial Interpolation
5. Bicubic Spline Interpolation
6. Multiple Linear and Quadratic Regressio
n
7. Image Interpolation
8. Exit
Masukkan pilihan (1-8): 1
1. Gauss Elimination Method
2. Gauss-Jordan Elimination Method
3. Inverse Matrix Method
4. Cramer's Rule
Masukkan pilihan (1-4): 1
Masukkan metode penginputan:
1. Input manual
2. Input dari file .txt
Masukkan pilihan (1-2): 2
Masukkan nama file: SPL/1.txt
SPL tidak memiliki solusi.
```

- Metode Gauss Jordan:



```
+=========================================+
|                  MENU                   |
+=========================================+

1. System of Linear Equations
2. Determinant
3. Inverse Matrix
4. Polynomial Interpolation
5. Bicubic Spline Interpolation
6. Multiple Linear and Quadratic Regressio
n
7. Image Interpolation
8. Exit
Masukkan pilihan (1-8): 1
1. Gauss Elimination Method
2. Gauss-Jordan Elimination Method
3. Inverse Matrix Method
4. Cramer's Rule
Masukkan pilihan (1-4): 2
Masukkan metode penginputan:
1. Input manual
2. Input dari file .txt
Masukkan pilihan (1-2): 2
Masukkan nama file: SPL/1.txt
x1 = -9.223372036854775E11
x2 = 9.223372036854775E11
x3 = 9.223372036854775E11
x4 = 9.223372036854775E11
```

- Metode Matriks Balikan:



```
+=========================================+
|                  MENU                   |
+=========================================+

1. System of Linear Equations
2. Determinant
3. Inverse Matrix
4. Polynomial Interpolation
5. Bicubic Spline Interpolation
6. Multiple Linear and Quadratic Regressio
n
7. Image Interpolation
8. Exit
Masukkan pilihan (1-8): 1
1. Gauss Elimination Method
2. Gauss-Jordan Elimination Method
3. Inverse Matrix Method
4. Cramer's Rule
Masukkan pilihan (1-4): 3
Masukkan metode penginputan:
1. Input manual
2. Input dari file .txt
Masukkan pilihan (1-2): 2
Masukkan nama file: SPL/1.txt
x1 = -9.223372036854775E11
x2 = 9.223372036854775E11
x3 = 9.223372036854775E11
x4 = 9.223372036854775E11
```

- Kaidah Cramer:

```
+=========================================+
|                  MENU                   |
+=========================================+

1. System of Linear Equations
2. Determinant
3. Inverse Matrix
4. Polynomial Interpolation
5. Bicubic Spline Interpolation
6. Multiple Linear and Quadratic Regressio
n
7. Image Interpolation
8. Exit
Masukkan pilihan (1-8): 1
1. Gauss Elimination Method
2. Gauss-Jordan Elimination Method
3. Inverse Matrix Method
4. Cramer's Rule
Masukkan pilihan (1-4): 4
Masukkan metode penginputan:
1. Input manual
2. Input dari file .txt
Masukkan pilihan (1-2): 2
Masukkan nama file: SPL/1.txt
Kesalahan: Determinan matriks A bernilai 0
.
Tidak dapat menemukan solusi SPL.
```

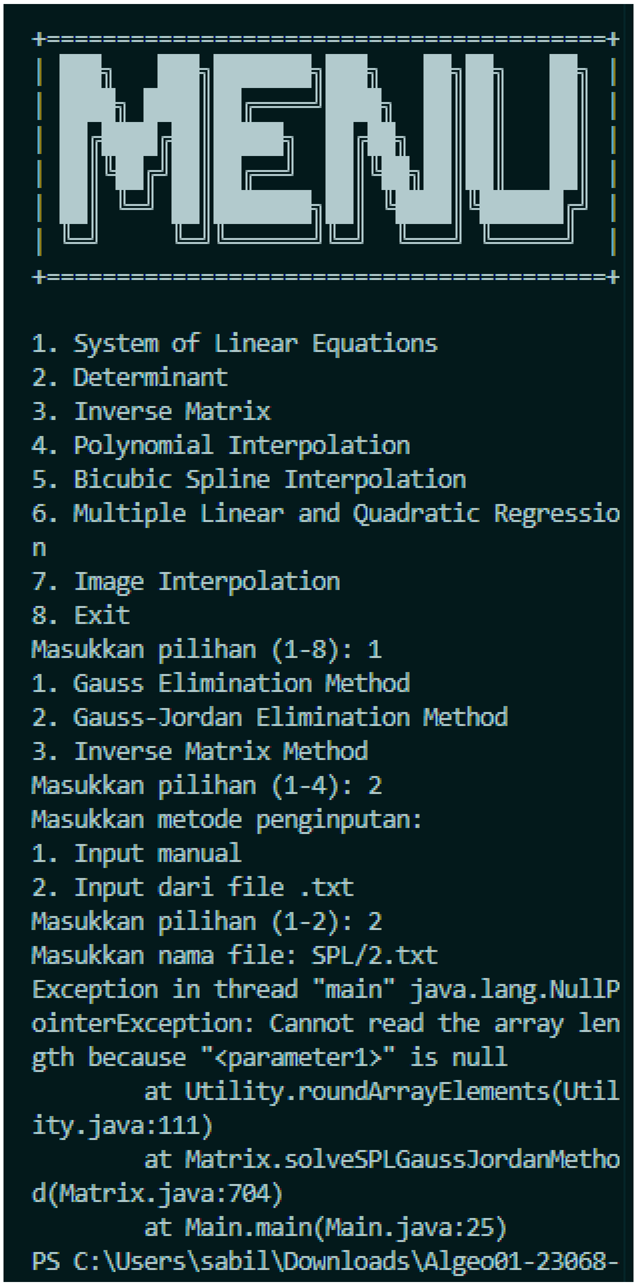
b. Test case 2

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

- Metode Gauss:

```
MENU

1. System of Linear Equations
2. Determinant
3. Inverse Matrix
4. Polynomial Interpolation
5. Bicubic Spline Interpolation
6. Multiple Linear and Quadratic Regressio
n
7. Image Interpolation
8. Exit
Masukkan pilihan (1-8): 1
1. Gauss Elimination Method
2. Gauss-Jordan Elimination Method
3. Inverse Matrix Method
4. Cramer's Rule
Masukkan pilihan (1-4): 1
Masukkan metode penginputan:
1. Input manual
2. Input dari file .txt
Masukkan pilihan (1-2): 2
Masukkan nama file: SPL/2.txt
x1 = 3.0 + b
x2 = 2.0b
x3 = a
x4 = -1.0 + b
x5 = b
```

- Metode Gauss Jordan:



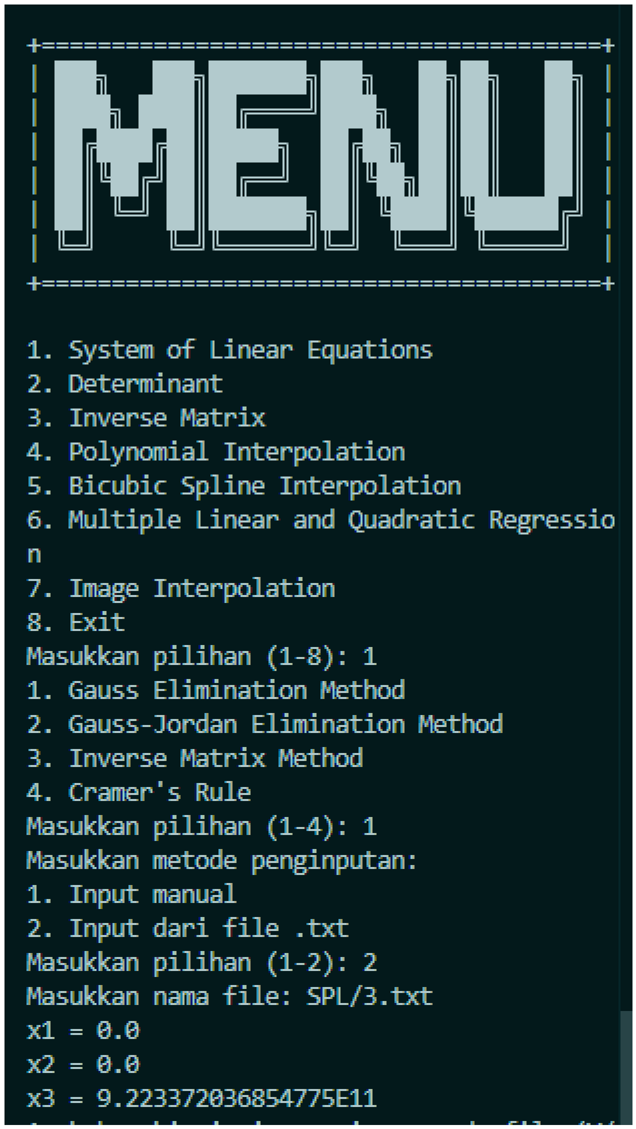
```
+=========================================+
|                                         |
+=========================================+

1. System of Linear Equations
2. Determinant
3. Inverse Matrix
4. Polynomial Interpolation
5. Bicubic Spline Interpolation
6. Multiple Linear and Quadratic Regressio
n
7. Image Interpolation
8. Exit
Masukkan pilihan (1-8): 1
1. Gauss Elimination Method
2. Gauss-Jordan Elimination Method
3. Inverse Matrix Method
Masukkan pilihan (1-4): 2
Masukkan metode penginputan:
1. Input manual
2. Input dari file .txt
Masukkan pilihan (1-2): 2
Masukkan nama file: SPL/2.txt
Exception in thread "main" java.lang.NullP
ointerException: Cannot read the array len
gth because "<parameter1>" is null
        at Utility.roundArrayElements(Util
ity.java:111)
        at Matrix.solveSPLGaussJordanMetho
d(Matrix.java:704)
        at Main.main(Main.java:25)
PS C:\Users\sabil\Downloads\Algeo01-23068-
```

- Metode Matriks Balikan:
  Jumlah persamaan tidak sama dengan jumlah variabel
- Kaidah Cramer:
  Jumlah persamaan tidak sama dengan jumlah variabel

c. Test case 3

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

- Metode Gauss:



```
1. System of Linear Equations
2. Determinant
3. Inverse Matrix
4. Polynomial Interpolation
5. Bicubic Spline Interpolation
6. Multiple Linear and Quadratic Regressio
n
7. Image Interpolation
8. Exit
Masukkan pilihan (1-8): 1
1. Gauss Elimination Method
2. Gauss-Jordan Elimination Method
3. Inverse Matrix Method
4. Cramer's Rule
Masukkan pilihan (1-4): 1
Masukkan metode penginputan:
1. Input manual
2. Input dari file .txt
Masukkan pilihan (1-2): 2
Masukkan nama file: SPL/3.txt
x1 = 0.0
x2 = 0.0
x3 = 9.223372036854775E11
```

- Metode Gauss Jordan:

```
+========================================+
|                 MENU                   |
+========================================+

1. System of Linear Equations
2. Determinant
3. Inverse Matrix
4. Polynomial Interpolation
5. Bicubic Spline Interpolation
6. Multiple Linear and Quadratic Regressio
n
7. Image Interpolation
8. Exit
Masukkan pilihan (1-8): 1
1. Gauss Elimination Method
2. Gauss-Jordan Elimination Method
3. Inverse Matrix Method
4. Cramer's Rule
Masukkan pilihan (1-4): 2
Masukkan metode penginputan:
1. Input manual
2. Input dari file .txt
Masukkan pilihan (1-2): 2
Masukkan nama file: SPL/3.txt
x1 = 1.0
x2 = -2.0
x3 = 1.0
```

- Metode Matriks Balikan:
  Jumlah persamaan tidak sama dengan jumlah variabel
- Kaidah Cramer:
  Jumlah persamaan tidak sama dengan jumlah variabel

d. Test case 4

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

Kasus n = 6:

- Metode Gauss:

- Metode Gauss Jordan:

- Metode Matriks Balikan:

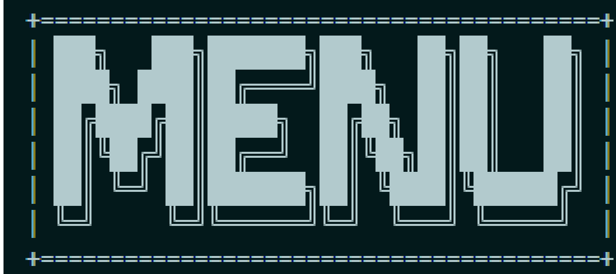
- Kaidah Cramer:

4.2. SPL berbentuk matriks augmented

a. Test case 1

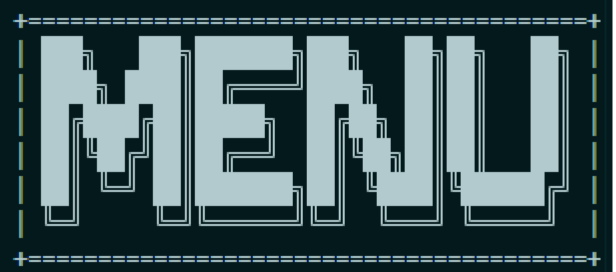
$$\begin{bmatrix} 1 & -1 & 2 & -1 & -1 \\ 2 & 1 & -2 & -2 & -2 \\ -1 & 2 & -4 & 1 & 1 \\ 3 & 0 & 0 & -3 & -3 \end{bmatrix}.$$

- Metode Gauss:



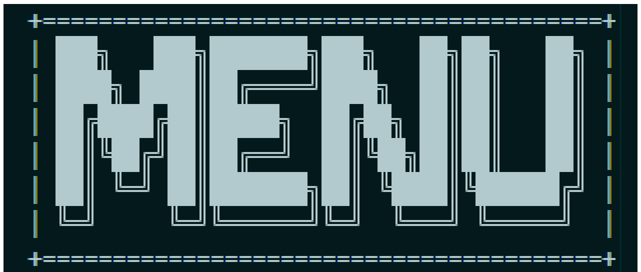
```
1. System of Linear Equations
2. Determinant
3. Inverse Matrix
4. Polynomial Interpolation
5. Bicubic Spline Interpolation
6. Multiple Linear and Quadratic Regressio
n
7. Image Interpolation
8. Exit
Masukkan pilihan (1-8): 1
1. Gauss Elimination Method
2. Gauss-Jordan Elimination Method
3. Inverse Matrix Method
4. Cramer's Rule
Masukkan pilihan (1-4): 1
Masukkan metode penginputan:
1. Input manual
2. Input dari file .txt
Masukkan pilihan (1-2): 2
Masukkan nama file: SPL/4.txt
x1 = -1.0 + b
x2 = 2.0a
x3 = a
x4 = b
```

- Metode Gauss Jordan:



```
1. System of Linear Equations
2. Determinant
3. Inverse Matrix
4. Polynomial Interpolation
5. Bicubic Spline Interpolation
6. Multiple Linear and Quadratic Regressio
n
7. Image Interpolation
8. Exit
Masukkan pilihan (1-8): 1
1. Gauss Elimination Method
2. Gauss-Jordan Elimination Method
3. Inverse Matrix Method
4. Cramer's Rule
Masukkan pilihan (1-4): 2
Masukkan metode penginputan:
1. Input manual
2. Input dari file .txt
Masukkan pilihan (1-2): 2
Masukkan nama file: SPL/4.txt
Exception in thread "main" java.lang.NullP
ointerException: Cannot read the array len
gth because "<parameter1>" is null
        at Utility.roundArrayElements(Util
ity.java:111)
        at Matrix.solveSPLGaussJordanMetho
d(Matrix.java:704)
        at Main.main(Main.java:25)
```

- Metode Matriks Balikan:




```
1. System of Linear Equations
2. Determinant
3. Inverse Matrix
4. Polynomial Interpolation
5. Bicubic Spline Interpolation
6. Multiple Linear and Quadratic Regressio
n
7. Image Interpolation
8. Exit
Masukkan pilihan (1-8): 1
1. Gauss Elimination Method
2. Gauss-Jordan Elimination Method
3. Inverse Matrix Method
4. Cramer's Rule
Masukkan pilihan (1-4): 3
Masukkan metode penginputan:
1. Input manual
2. Input dari file .txt
Masukkan pilihan (1-2): 2
Masukkan nama file: SPL/4.txt
x1 = 0.0
x2 = 0.0
x3 = 0.0
x4 = 0.0
```

- Kaidah Cramer:

```
+========================================+
|                 MENU                   |
+========================================+

1. System of Linear Equations
2. Determinant
3. Inverse Matrix
4. Polynomial Interpolation
5. Bicubic Spline Interpolation
6. Multiple Linear and Quadratic Regressio
n
7. Image Interpolation
8. Exit
Masukkan pilihan (1-8): 1
1. Gauss Elimination Method
2. Gauss-Jordan Elimination Method
3. Inverse Matrix Method
4. Cramer's Rule
Masukkan pilihan (1-4): 4
Masukkan metode penginputan:
1. Input manual
2. Input dari file .txt
Masukkan pilihan (1-2): 2
Masukkan nama file: SPL/4.txt
Kesalahan: Determinan matriks A bernilai 0
.
Tidak dapat menemukan solusi SPL.
```

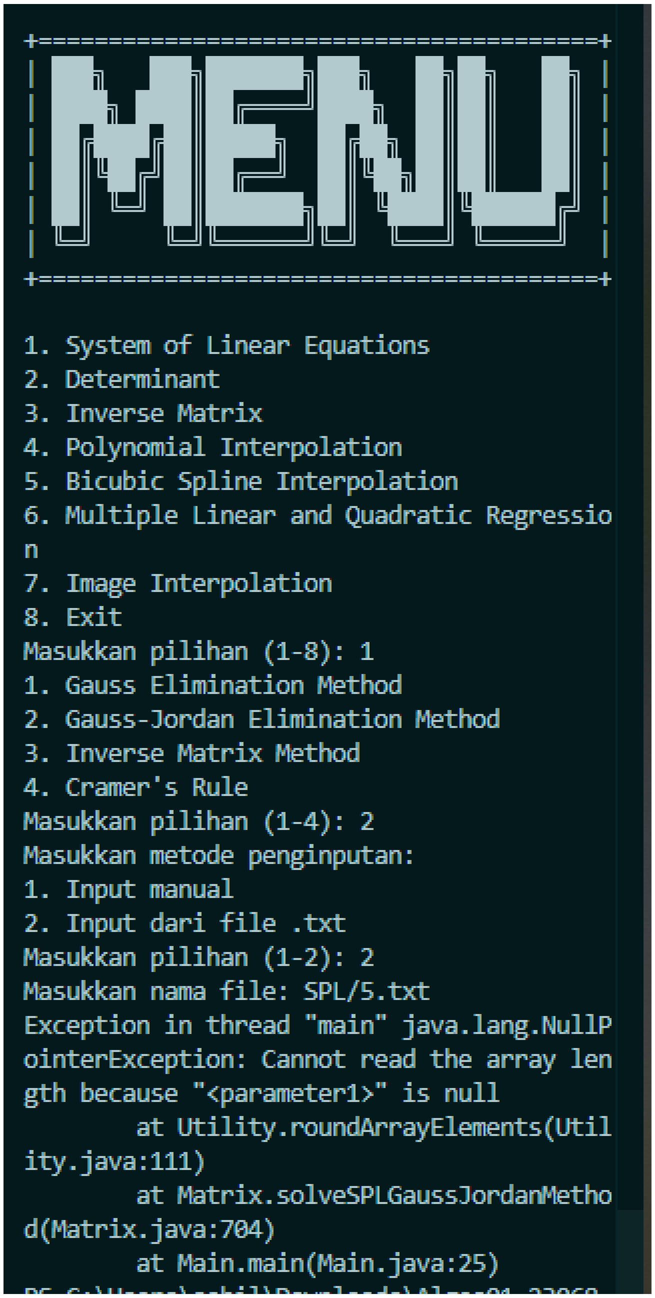
b. Test case 2

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

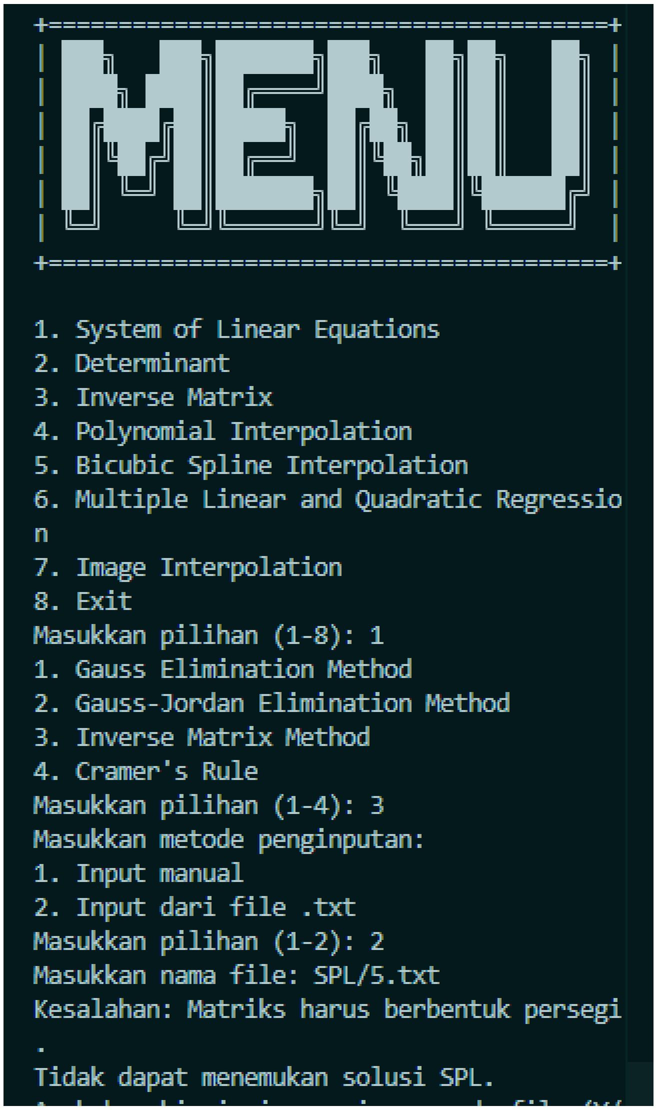
- Metode Gauss:

```
1. System of Linear Equations
2. Determinant
3. Inverse Matrix
4. Polynomial Interpolation
5. Bicubic Spline Interpolation
6. Multiple Linear and Quadratic Regressio
n
7. Image Interpolation
8. Exit
Masukkan pilihan (1-8): 1
1. Gauss Elimination Method
2. Gauss-Jordan Elimination Method
3. Inverse Matrix Method
4. Cramer's Rule
Masukkan pilihan (1-4): 1
Masukkan metode penginputan:
1. Input manual
2. Input dari file .txt
Masukkan pilihan (1-2): 2
Masukkan nama file: SPL/5.txt
x1 = 0
x2 = 2.0
x3 = 1.0
x4 = 1.0
```

- Metode Gauss Jordan:

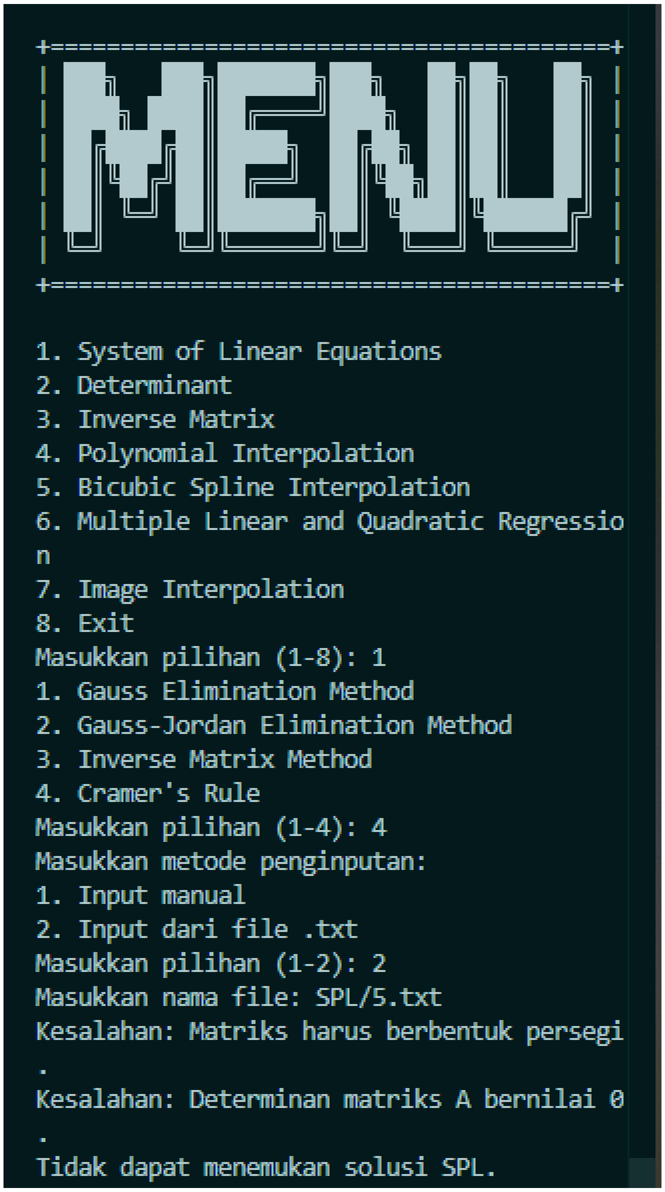


- Metode Matriks Balikan:



```
1. System of Linear Equations
2. Determinant
3. Inverse Matrix
4. Polynomial Interpolation
5. Bicubic Spline Interpolation
6. Multiple Linear and Quadratic Regressio
n
7. Image Interpolation
8. Exit
Masukkan pilihan (1-8): 1
1. Gauss Elimination Method
2. Gauss-Jordan Elimination Method
3. Inverse Matrix Method
4. Cramer's Rule
Masukkan pilihan (1-4): 3
Masukkan metode penginputan:
1. Input manual
2. Input dari file .txt
Masukkan pilihan (1-2): 2
Masukkan nama file: SPL/5.txt
Kesalahan: Matriks harus berbentuk persegi
.
Tidak dapat menemukan solusi SPL.
```

- Kaidah Cramer:



```
1. System of Linear Equations
2. Determinant
3. Inverse Matrix
4. Polynomial Interpolation
5. Bicubic Spline Interpolation
6. Multiple Linear and Quadratic Regressio
n
7. Image Interpolation
8. Exit
Masukkan pilihan (1-8): 1
1. Gauss Elimination Method
2. Gauss-Jordan Elimination Method
3. Inverse Matrix Method
4. Cramer's Rule
Masukkan pilihan (1-4): 4
Masukkan metode penginputan:
1. Input manual
2. Input dari file .txt
Masukkan pilihan (1-2): 2
Masukkan nama file: SPL/5.txt
Kesalahan: Matriks harus berbentuk persegi
.
Kesalahan: Determinan matriks A bernilai 0
.
Tidak dapat menemukan solusi SPL.
```

4.3. SPL berbentuk

a. Test case 1

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

- Metode Gauss:

- Metode Gauss Jordan:

- Metode Matriks Balikan:

- Kaidah Cramer:

b. Test case 2

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

- Metode Gauss:

- Metode Gauss Jordan:

- Metode Matriks Balikan:

- Kaidah Cramer:

4.4. Studi Kasus Interpolasi

a. Test case 1

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

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

Lakukan pengujian pada nilai-nilai default berikut:

- x = 0.2, f(x) = ?

- x = 0.55, f(x) = ?

- x = 0.85, f(x) = ?

- x = 1.28, f(x) = ?

b. Test case 2

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

Hasil :

```
x1 = 0.6762
x2 = 0.2266
x3 = -0.0064
Persamaan interpolasi:
y = 0.6762 + 0.2266x^1 - 0.0064x^2
Taksiran nilai y pada x = 8.3 adalah 2.1160840000000003
```

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

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

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

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

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

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

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

- 16/07/2022 (7.516)
- 10/08/2022 (8.323)
- 05/09/2022 (9.167)
- 03/10/2022 (10.097)

c. Test case 3

Sederhanakan fungsi

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

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

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

4.5 Studi Kasus Interpolasi Bicubic

```
21 98 125 153
52 101 161 59
0 42 72 320
16 12 81 96
```

a. TC-Bicubic-1.txt

```
┌──(andrewlinux㉿Andrew)-[~/CodeMe/Tubes_Algeo_01/Kelompok20Tubes1Algeo/Algeo01-23068]
└─$ java -cp bin BicubicSplineInterpolation
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=on -Dswing.aatext=true
Pilih opsi input:
1. Input manual
2. Input dari file TXT
2
Masukkan nama file (dengan ekstensi .txt):
test7.1.txt
Nilai interpolasi di titik (0.0, 0.0): 21.0
```

b. TC-Bicubic-2.txt

```
┌──(andrewlinux㉿Andrew)-[~/CodeMe/Tubes_Algeo_01/Kelompok20Tubes1Algeo/Algeo01-23068]
└─$ java -cp bin BicubicSplineInterpolation
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=on -Dswing.aatext=true
Pilih opsi input:
1. Input manual
2. Input dari file TXT
2
Masukkan nama file (dengan ekstensi .txt):
test7.2.txt
Nilai interpolasi di titik (0.5, 0.5): 80.984375
```

c. TC-Bicubic-3.txt

```
┌──(andrewlinux㉿Andrew)-[~/CodeMe/Tubes_Algeo_01/Kelompok20Tubes1Algeo/Algeo01-23068]
└─$ java -cp bin BicubicSplineInterpolation
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=on -Dswing.aatext=true
Pilih opsi input:
1. Input manual
2. Input dari file TXT
2
Masukkan nama file (dengan ekstensi .txt):
test7.3.txt
Nilai interpolasi di titik (0.25, 0.75): 115.337158203125
```

d. TC-Bicubic-4.txt

```
┌──(andrewlinux㉿Andrew)-[~/CodeMe/Tubes_Algeo_01/Kelompok20Tubes1Algeo/Algeo01-23068]
└─$ java -cp bin BicubicSplineInterpolation
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=on -Dswing.aatext=true
Pilih opsi input:
1. Input manual
2. Input dari file TXT
2
Masukkan nama file (dengan ekstensi .txt):
test7.4.txt
Nilai interpolasi di titik (0.1, 0.9): 128.32797499999998
```

4.6 Studi Kasus Linear Berganda

Diberikan sekumpulan data sesuai pada tabel berikut ini.

Table 12.1: Data for Example 12.1

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

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

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

Persamaan yang didapatkan:

```
Persamaan Regresi Linear Berganda yang Memenuhi adalah berikut.
Y = -3.5114699 - 0.0026244X1 + 7.989E-4X2 + 0.1542797X3
Hampiran dari data adalah Y = 0.9384217100000005
```

Hampiran nilai y dari data yang diinginkan:

```
Persamaan Regresi Linear Berganda yang Memenuhi adalah berikut.
Y = -3.5114699 - 0.0026244X1 + 7.989E-4X2 + 0.1542797X3
Hampiran dari data adalah Y = 0.9384217100000005
```

4.7. Studi kasus pencarian determinan

a. Test case 1

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

-Metode reduksi baris

```
1. Cofactor Method
2. Gauss Elimination Method
Masukkan pilihan (1-2): 2
Masukkan metode penginputan:
1. Input manual
2. Input dari file .txt
Masukkan pilihan (1-2): 2
Masukkan nama file: Det1.txt
Determinan matriks: 6781745.99999999
```

-Metode ekspansi kofaktor

```
1. Cofactor Method
2. Gauss Elimination Method
Masukkan pilihan (1-2): 1
Masukkan metode penginputan:
1. Input manual
2. Input dari file .txt
Masukkan pilihan (1-2): 2
Masukkan nama file: Det1.txt
Determinan matriks: 6781746.0
```

b. Test case 2

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

-Metode reduksi baris

```
Masukkan nama file: Det2.txt
Determinan matriks: 8.0
```

-Metode ekspansi kofaktor

```
Masukkan nama file: Det2.txt
Determinan matriks: 8.0
```

Untuk beberapa test case lainnya, akan dibuktikan pendemoan

BAB 5

KESIMPULAN

5.1. Kesimpulan

Berbagai metode dapat digunakan dalam menyelesaikan Sistem Persamaan Linear (SPL), pencarian determinan, dan penentuan matriks invers. Di antara metode penyelesaian SPL yang umum digunakan adalah metode eliminasi Gauss, metode Gauss-Jordan, metode matriks invers, dan kaidah Cramer. Untuk pencarian determinan, terdapat beberapa pendekatan, termasuk metode reduksi baris dan metode ekspansi kofaktor.

Metode-metode tersebut memiliki aplikasi yang luas dan beragam dalam pemecahan masalah matematis. Beberapa aplikasi yang signifikan meliputi ekspansi bikubik, interpolasi polinom, regresi linier berganda, dan regresi kuadratik berganda. Dengan pemahaman dan penerapan yang tepat, metode-metode ini tidak hanya membantu dalam menyelesaikan masalah matematis, tetapi juga memainkan peran penting dalam berbagai bidang, seperti statistik, analisis data, dan pemodelan matematis. Hal ini menunjukkan betapa fundamentalnya pemahaman tentang metode penyelesaian SPL dan determinan dalam kajian ilmu terapan.

5.2. Saran

Beberapa saran pengembangan dari program ini adalah sebagai berikut:

- Penggunaan *rounding* untuk menghasilkan hasil kalkulasi yang lebih akurat.
- Dalam menentukan regresi, dapat dilakukan perkalian dengan transpose dari matriks x, agar hasil regresi lebih baik.
- Sebaiknya ditentukan satu solusi dari solusi parametrik agar nilai hampiran Y dapat ditentukan.

5.3. Refleksi

Pada awalnya pengerjaan tubes ini cukup lancar, hanya saja ditengah proses pengerjaannya terdapat beberapa kendala disebabkan banyaknya agenda yang kami lakukan. Walaupun tugas ini dapat terselesaikan, tapi kami merasa masih banyak aspkek yang dapat ditingkatkan.

DAFTAR PUSTAKA

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

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

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

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

Link repository: https://github.com/bill2247/Algeo01-23068