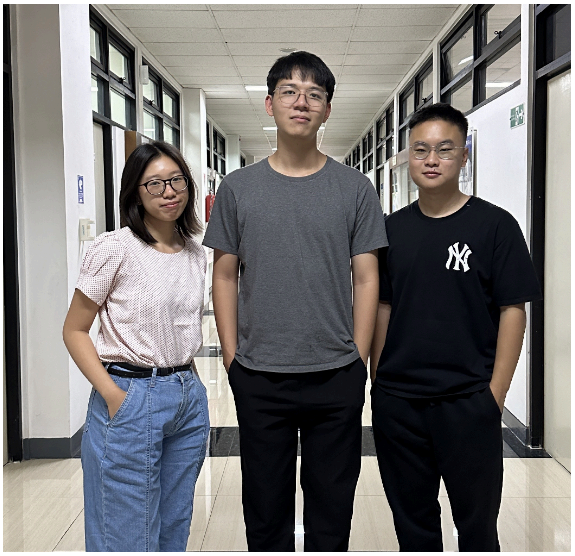
LAPORAN TUGAS BESAR 1 IF2123 ALJABAR LINEAR DAN GEOMETRI

SISTEM PERSAMAAN LINEAR, DETERMINAN, DAN APLIKASINYA



Dosen Pengampu	: Ir. Rila Mandala, M.Eng., Ph.D.
Asisten Pembimbing	: M. Fauzan Azhim (13522153)

Disusun oleh:

Kelompok 46 - Silahkan Siapapun Yang Mau Sekelompok Samaku (SSYMSS)

Nathaniel Jonathan Rusli	(13523013)
Benedict Presley	(13523067)
Naomi Risaka Sitorus	(13523122)

SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA

INSTITUT TEKNOLOGI BANDUNG

2023

KATA PENGANTAR

Dengan penuh rasa syukur, kami mempersembahkan makalah ini yang membahas materi penting dalam bidang matematika dan pemrograman, yaitu "Aljabar Linier dan Geometri". Makalah ini disusun sebagai bagian dari tugas mata kuliah IF2123 Aljabar Linier dan Geometri yang memiliki peran krusial dalam fondasi matematika modern serta penerapannya di berbagai disiplin ilmu.

Mata kuliah ini tidak hanya mengajarkan konsep dasar yang esensial, tetapi juga memperkenalkan metode dan teknik analitik yang berperan signifikan dalam pemecahan masalah yang lebih kompleks. Pemahaman yang baik tentang Aljabar Linier dan Geometri sangat penting dalam dunia yang semakin *data-driven*, karena konsep-konsep ini sering diaplikasikan dalam analisis data, optimasi, grafik komputer, kecerdasan buatan, dan bidang lainnya.

Melalui makalah ini, kami berharap dapat memberikan gambaran yang komprehensif dan mendalam tentang topik yang dibahas, dengan harapan dapat membantu pembaca untuk tidak hanya memahami teori dasar, tetapi juga melihat relevansi praktisnya dalam dunia nyata. Kami juga berharap makalah ini bisa menjadi panduan yang bermanfaat bagi para pembaca yang ingin memperluas wawasan mereka dalam bidang yang menarik ini.

Kami ucapkan terima kasih kepada dosen pembimbing dan semua pihak yang telah memberikan dukungan, bimbingan, dan masukan dalam proses penyusunan makalah ini. Semoga makalah ini dapat memberikan manfaat yang nyata dan mendorong semangat eksplorasi lebih lanjut dalam memahami Aljabar Linier dan Geometri.

Akhir kata, kami berharap makalah ini dapat memotivasi pembaca untuk terus mendalami dan mengeksplorasi lebih jauh dunia matematika yang penuh tantangan dan peluang ini.

Bandung, 24 Oktober 2024

Nathaniel Jonathan Rusli	(13523013)
Benedict Presley	(13523067)
Naomi Risaka Sitorus	(13523122)

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

DAFTAR PEMBAGIAN TUGAS

Tabel 1. Pembagian Tugas Pembuatan Tugas Besar Kelompok 46

Kegiatan		PIC
ADT	ADT *Matrix*	13523013 13523067 13523122
Library	Eliminasi Gauss	13523067
	Eliminasi Gauss-Jordan	13523013
	Determinan (Reduksi Baris)	13523067
	Determinan (Kofaktor)	13523067
	Hitung Kofaktor	13523067
	Matriks Balikan (Adjoin/Determinan)	13523122
	Matriks Balikan (Gauss Jordan Elimination)	13523013
	SPL (Eliminasi Gauss dan Matriks Balikan)	13523122
	SPL (Eliminasi Gauss-Jordan)	13523013
	SPL (Matriks Balikan)	13523122
	SPL (Cramer)	13523013
Program	Interpolasi Polinomial	13523122
	Regresi Linear Berganda	13523013 13523122
	Regresi Kuadratik Berganda	13523013
	Bicubic Spline Interpolation	13523067
	Interpolasi Gambar	13523067
	Main Program	13523122
	File Operations	13523067 13523122
Laporan	Bab 1 (Deskripsi Masalah)	13523013

	Bab 2 (Teori Singkat)	13523013
	Bab 3 (Implementasi Pustaka dan Program)	13523013 13523067 13523122
	Bab 4 (Eksperimen)	13523013 13523067 13523122
	Bab 5 (Kesimpulan, Saran, Refleksi)	13523013 13523067 13523122

BAB I
DESKRIPSI MASALAH

1.1 Sistem Persamaan Linear (SPL)

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

$$\begin{bmatrix} 0 & \mathbf{2} & 1 & -1 \\ 0 & 0 & \mathbf{3} & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}. \quad \begin{bmatrix} 0 & \mathbf{1} & 0 & -\frac{2}{3} \\ 0 & 0 & \mathbf{1} & \frac{1}{3} \\ 0 & 0 & 0 & 0 \end{bmatrix}.$$

Gambar 1.1.1 Eliminasi Gauss dilakukan dengan matriks eselon baris dan eliminasi Gauss-Jordan dengan matriks eselon baris tereduksi.

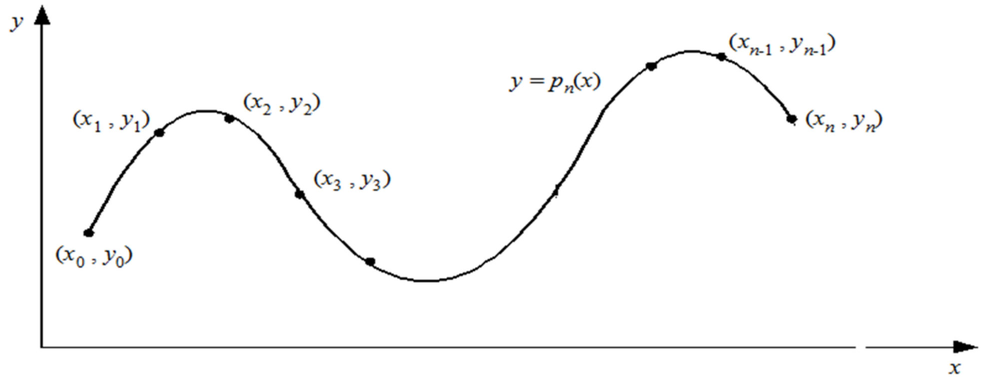
Di dalam Tugas Besar 1 ini, Anda diminta membuat satu atau lebih *library* aljabar linier dalam Bahasa Java. Library tersebut berisi fungsi-fungsi seperti eliminasi Gauss, eliminasi Gauss-Jordan, menentukan balikan matriks, menghitung determinan, kaidah Cramer (kaidah Cramer khusus untuk SPL dengan n peubah dan n persamaan). Selanjutnya, gunakan *library* tersebut di dalam program Java untuk menyelesaikan berbagai persoalan yang dimodelkan dalam bentuk SPL, menyelesaikan persoalan interpolasi, dan persoalan regresi. Penjelasan tentang interpolasi dan regresi adalah seperti di bawah ini.

Beberapa tulisan cara membuat library di Java:

1. https://www.programcreek.com/2011/07/build-a-java-library-for-yourself/
2. https://developer.ibm.com/tutorials/j-javalibrary/

1.2 Interpolasi Polinomial

Persoalan interpolasi polinom adalah sebagai berikut: Diberikan n+1 buah titik berbeda, (x_0, y_0), (x_1, y_1), ..., (x_n, y_n). Tentukan 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, …, n.



Gambar 2.1.1 Ilustrasi beberapa titik yang diinterpolasi secara polinomial.

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), ..., (x_n, y_n). adalah berbentuk $p_n(x) = a_0 + a_1x + a_2x^2 + \ldots + a_nx^n$. 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 persaman 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 $p_n(x) = a_0 + a_1x + a_2x^2 + \ldots + a_nx^n$ untuk i = 0, 1, 2, …, n, akan diperoleh n buah sistem persamaan lanjar dalam a_0, a_1, a_2, …, a_n,

$$
\begin{aligned}
a_0 + a_1x_0 + a_2x_0^2 + \ldots + a_n x_0^n &= y_0 \\
a_0 + a_1x_1 + a_2x_1^2 + \ldots + a_n x_1^n &= y_1 \\
\ldots \qquad & \ldots \\
a_0 + a_1x_n + a_2x_n^2 + \ldots + a_n x_n^n &= y_n
\end{aligned}
$$

Solusi sistem persamaan lanjar ini, yaitu nilai a_0, a_1, …, a_n, diperoleh dengan menggunakan metode eliminasi Gauss yang sudah anda pelajari. Sebagai contoh, misalkan diberikan tiga buah titik yaitu (8.0, 2.0794), (9.0, 2.1972), dan (9.5, 2.2513). Tentukan polinom interpolasi kuadratik lalu estimasi nilai fungsi pada x = 9.2. Polinom kuadratik berbentuk $p_2(x) = a_0 + a_1x + a_2x^2$. Dengan menyulihkan ketiga buah titik data ke dalam polinom tersebut, diperoleh sistem persamaan lanjar yang terbentuk adalah

$$a_0 + 8.0a_1 + 64.00a_2 = 2.0794$$

$$a_0 + 9.0a_1 + 81.00a_2 = 2.1972$$

$$a_0 + 9.5a_1 + 90.25a_2 = 2.2513$$

Penyelesaian sistem persamaan dengan metode eliminasi Gauss menghasilkan a_0 = 0.6762, a_1 = 0.2266, dan a_2 = -0.0064. Polinom interpolasi yang melalui ketiga buah titik tersebut adalah $p_2(x) = 0.6762 + 0.2266x - 0.0064x^2$. Dengan menggunakan polinom ini, maka nilai fungsi pada x = 9.2 dapat ditaksir sebagai berikut: $p_2(9.2) = 0.6762 + 0.2266(9.2) - 0.0064(9.2)^2 = 2.2192$.

1.3 Regresi Berganda

Regresi (akan dipelajari lebih lanjut di Probabilitas dan Statistika) merupakan salah satu metode untuk memprediksi nilai selain menggunakan Interpolasi Polinom. Pada tugas besar ini, anda diminta untuk membuat 2 jenis regresi yaitu Regresi Linier Berganda dan Regresi Kuadratik Berganda.

1.3.1 Regresi Linier Berganda

Meskipun sudah ada persamaan jadi untuk menghitung regresi linear sederhana, terdapat persamaan umum dari regresi linear yang bisa digunakan untuk regresi linear berganda, yaitu.

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

Gambar 2.3.1.1 Persamaan Umum Regresi Linear

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

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

Gambar 2.3.1.2 *Normal Estimation Equation for Multiple Linear Regression*

1.3.2 Regresi Kuadratik Berganda

Dalam kasus ini, proses mengubah data-data dalam regresi kuadratik berganda cukup berbeda dengan Regresi Linier Berganda. Bentuk persamaan dari regresi kuadratik ada 3, yaitu:

a. Variabel Linier: Variabel dengan derajat satu seperti X, Y, dan Z
b. Variabel Kuadrat: Variabel dengan derajat dua seperti X2
c. 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. Perhatikan contoh regresi kuadratik 2 variabel peubah 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 2.3.2.1 Matriks Regresi Kuadratik Dua Variabel

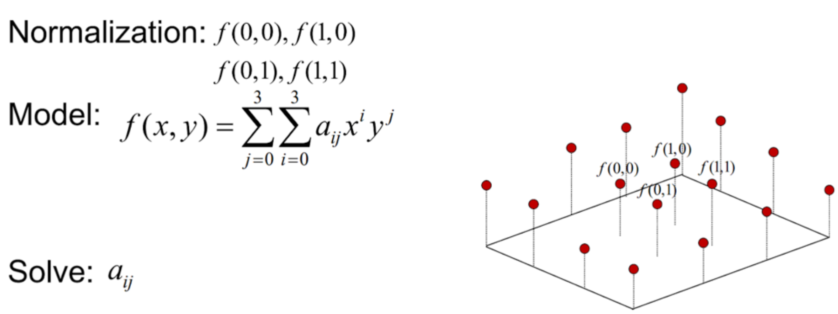
N menandakan jumlah peubah, terdapat 2 variabel linier yaitu ui dan vi, 2 variabel kuadrat yaitu ui2 dan vi2, dan 1 variabel interaksi yaitu uv. Untuk setiap n-peubah, akan terdapat 1 konstan N (Terlihat di bagian atas kiri gambar), n variabel linier, n variabel kuadrat, dan C2n variabel linier (dengan 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.

Kedua model regresi yang dijadikan sistem persamaan linier tersebut diselesaikan dengan menggunakan metode eliminasi Gauss.

1.4 *Bicubic Spline Interpolation*

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 membagun permukaan bikubik. Bentuk pemodelannya adalah sebagai berikut.



Gambar 2.4.1 Pemodelan interpolasi *bicubic spline*.

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

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

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

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

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

Gambar 2.4.2 Persamaan Polinomial *Bicubic Spline Interpolation*

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

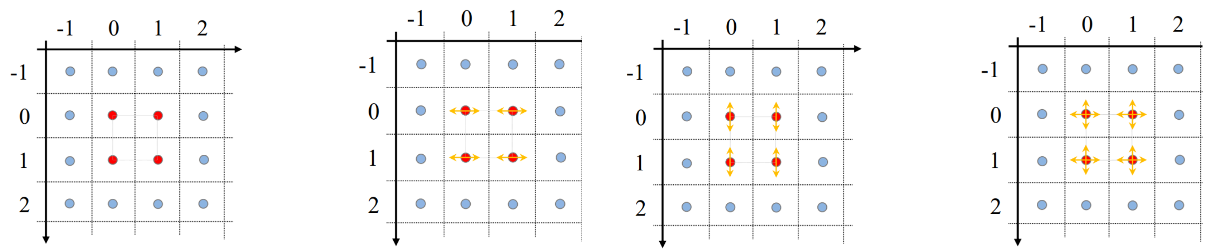
$$y = Xa$$

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

Gambar 2.4.3 Persamaan Penyelesaian Bicubic Spline Interpolation

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 x 1^{1-1} x 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. Tugas Anda pada studi kasus ini adalah membangun persamaan $f(x, y)$ yang akan digunakan untuk melakukan interpolasi berdasarkan nilai $f(a, b)$ dari masukan matriks 4 x 4. Nilai masukan a dan b berada dalam rentang [0, 1]. Nilai yang akan diinterpolasi dan turunan berarah disekitarnya dapat diilustrasikan pada titik berwarna merah pada gambar di bawah.



Gambar 2.4.4 Nilai fungsi yang akan di interpolasi pada titik merah, turunan berarah terhadap sumbu x, terhadap sumbu y, dan keduanya (kiri ke kanan).

BAB II
TEORI SINGKAT

2.1 Eliminasi Gauss

Eliminasi gauss ditemukan oleh Carl Friedrich Gauss, metode ini dapat dimanfaatkan untuk memecahkan sistem persamaan linear dengan merepresentasikan (mengubah) menjadi bentuk matriks, matriks tersebut lalu diubah ke bentuk eselon baris melalui operasi baris elementer (OBE). Kemudian sistem diselesaikan dengan substitusi balik (Profematika, 2019).

Menurut Munir (2023), toga operasi baris elementer terhadap matriks augmented antara lain: pengalian sebuah baris dengan konstanta tidak nol, pertukaran dua buah baris, dan penambahan sebuah baris dengan kelipatan baris lainnya. Solusi sebuah SPL dapat diperoleh dengan menerapkan operasi baris elementer pada matriks *augmented* sampai terbentuk matriks eselon (metode eliminasi Gauss). Berikut adalah langkah metode eliminasi Gauss:

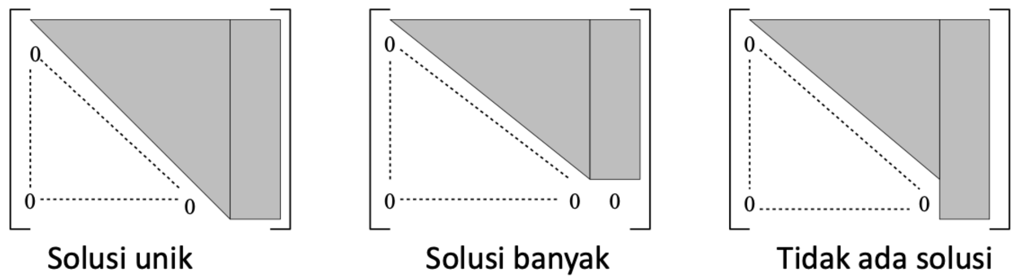
1. Nyatakan sistem persamaan linear dalam bentuk matriks *augmented*
2. Terapkan operasi baris elementer (OBE) pada matriks *augmented* sampai terbentuk matriks eselon baris

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

Gambar 2.1.1. Ilustrasi Penerapan OBE pada Matriks *Augmented* Hingga Terbentuk Matriks Eselon Baris

3. Selesaikan persamaan yang berkoresponden pada matriks eselon baris dengan teknik penyulihan mundur (*backward substitution).*

Terdapat tiga kemungkinan solusi yang dapat terjadi pada SPL, yaitu solusi yang unik (tunggal), banyak solusi (tak terhingga banyaknya), dan tidak memiliki solusi sama sekali. Solusi tunggal diperoleh bila satu utama matriks eselon baris terdapat pada diagonal matriks. Solusi banyak diperoleh ketika terdapat baris yang berisi seluruhnya 0 pada matriks *augmented* dan jumlah baris dengan satu utama yang lebih sedikit dari variabel peubahnya. Solusi yang tidak ada diperoleh ketika terdapat kontradiksi, yaitu baris pada matriks *augmented* yang berisi 0 kecuali pada kolom terakhir (Munir, 2023).



Gambar 2.1.2. Bentuk Matriks Akhir Yang Telah Dieliminasi dengan Metode Gauss dan Jenis Solusinya

2.2 Eliminasi Gauss-Jordan

Metode eliminasi Gauss-Jordan merupakan pengembangan metode eliminasi Gauss. Operasi baris elementer (OBE) diterapkan pada matriks *augmented* hingga menghasilkan matriks eselon tereduksi. Dengan menggunakan metode ini, tidak perlu lagi untuk melakukan substitusi secara mundur dalam mencari nilai-nilai variabel karena nilai variabel dapat diperoleh langsung dari matriks *augmented* akhir (bila solusinya unik) (Munir, 2024).

Metode eliminasi Gauss-Jordan terdiri dari dua fase yang dapat dilakukan secara bersamaan atau sekuensial:

1. Fase maju (*forward phase)* atau fase eliminasi Gauss: menghasilkan nilai-nilai 0 di bawah 1 utama)

$$\begin{bmatrix} 2 & 3 & -1 & 5 \\ 4 & 4 & -3 & 3 \\ -2 & 3 & -1 & 1 \end{bmatrix} \sim \overset{\text{OBE}}{\cdots} \sim \begin{bmatrix} 1 & 3/2 & -1/2 & 5/2 \\ 0 & 1 & 1/2 & 7/2 \\ 0 & 0 & 1 & 3 \end{bmatrix}$$

Gambar 2.2.1. *Forward Phase* Eliminasi Gauss-Jordan

2. Fase mundur (*backward phase)*: menghasilkan nilai-nilai 0 di atas satu utama

$$\begin{bmatrix} 1 & 3/2 & -1/2 & 5/2 \\ 0 & 1 & 1/2 & 7/2 \\ 0 & 0 & 1 & 3 \end{bmatrix} \overset{\text{R1} - (3/2)\text{R2}}{\sim} \begin{bmatrix} 1 & 0 & -5/4 & -11/4 \\ 0 & 1 & 1/2 & 7/2 \\ 0 & 0 & 1 & 3 \end{bmatrix} \underset{\text{R2 - (1/2)R3}}{\overset{\text{R1 +(5/4)R3}}{\sim}} \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 2 \\ 0 & 0 & 1 & 3 \end{bmatrix}$$

Gambar 2.2.2. *Backward Phase* Eliminasi Gauss-Jordan

2.3 Determinan

Menurut Rahma (2019), salah satu pembahasan dalam teori matriks adalah menentukan determinan suatu matriks. Determinan memiliki peranan penting dalam menyelesaikan beberapa persoalan dalam matriks dan banyak dipergunakan dalam ilmu matematika maupun ilmu terapannya. Nilai determinan matriks dapat menentukan invers matriks. Jika nilai determinan matriks tidak nol, maka matriks tersebut punya invers. Namun jika nilai determinannya nol, maka matriks tidak mempunyai invers. Nilai determinan juga dapat menyelesaikan sistem persamaan linier. Sistem persamaan linier ini banyak digunakan oleh bidang ilmu optimasi, ekonomi dan lainnya. Menghitung nilai determinan suatu matriks dapat menggunakan beberapa metode, diantaranya, metode Sarrus, metode ekspansi kofaktor, metode kondensasi chio, metode eliminasi gauss, dan metode dekomposisi.

2.4 Matriks Balikan

Matriks dapat dioperasikan seperti operasi yang dilakukan pada bilangan bulat, seperti penjumlahan dan perkalian. Selain itu, khusus matriks persegi, matriks dapat memiliki matriks inversnya. Matriks yang tidak memiliki balikan disebut sebagai matriks singular. Ide dari balikan matriks adalah ketika suatu matriks dikalikan dengan matriks A yang selanjutnya dikalikan lagi dengan balikan matriks A, maka akan dihasilkan matriks awal. Dalam matriks, suatu matriks yang dikalikan dengan matriks balikannya akan menghasilkan matriks identitas (Strang, 2006).

2.5 Matriks Kofaktor

Kofaktor adalah hasil perkalian minor dengan suatu angka yang besarnya menuruti suatu aturan yaitu $(-1)^{i+j}$, di mana i adalah baris dan j adalah kolom. Kofaktor suatu elemen baris ke-*i* dan kolom ke-*j* dari matriks A dilambangkan dengan:

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

Jumlah kofaktor suatu matriks mengikuti jumlah elemen matriks tersebut (Madematika, 2017).

Matriks kofaktor merupakan matriks yang terdiri dari kofaktor-kofaktor matriks itu sendiri. Misalkan terdapat suatu matriks A, maka matriks kofaktor A merupakan matriks yang terdiri dari kofaktor-kofaktor dari matriks A. Susunan elemen matriks kofaktor juga mengikuti susunan (letak) kofaktor-kofaktornya (Mademati ka, 2017).

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

Gambar 2.5.1. Matriks Kofaktor

2.6 Matriks Adjoin

Adjoin matriks merupakan transpose dari matriks kofaktor. Adjoin sering disingkat dengan Adj. Misalkan matriks A, maka adjoin A ditulis Adj (A). Transpose sendiri maksudnya adalah pertukaran elemen pada baris menjadi kolom atau kolom menjadi baris. Adjoin matriks digunakan dalam menentukan invers matriks (Madematika, 2017).

Balikan suatu matriks (misal matriks A) dapat dihitung dengan menggunakan rumus:

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

2.7 Kaidah Cramer

Dalam aljabar linear, kaidah Cramer adalah rumus yang dapat digunakan untuk menyelesaikan sistem persamaan linear dengan banyak persamaan sama dengan banyak variabel, dan berlaku ketika sistem tersebut memiliki solusi yang tunggal. Rumus ini menyatakan solusi dengan menggunakan determinan matriks koefisien (dari sistem persamaan) dan determinan matriks lain yang diperoleh dengan mengganti salah satu kolom matriks koefisien dengan vektor yang berada sebelah kanan persamaan (Wikipedia, 2017).

Kaidah Cramer yang digunakan dengan naif (apa adanya) tidak efisien secara komputasi untuk sistem dengan lebih dari dua atau tiga persamaan. Untuk kasus dengan n persamaan dalam n variabel, rumus ini perlu menghitung n + 1 nilai determinan, sedangkan eliminasi Gauss menghasilkan solusi yang sama dengan kompleksitas komputasi yang setara dengan menghitung satu nilai determinan. Kaidah Cramer juga dapat tidak stabil secara numerik bahkan untuk sistem ukuran 2×2. Namun, belakangan ini berhasil dibuktikan bahwa kaidah Cramer dapat diterapkan dalam kompleksitas waktu O(n3) (Wikipedia, 2017).

Pertimbangkan sistem persamaan linear dengan variabel, yang direpresentasikan dalam bentuk perkalian matriks sebagai:

$$Ax = b$$

dengan matriks berukuran memiliki determinan bukan nol, dan vektor $x = (x1, \ldots, xn)^T$ adalah vektor kolom dari variabel. Teorema menyatakan bahwa sistem memiliki solusi unik dalam keadaan ini, dengan nilai untuk setiap variabel diberikan oleh:

$$x_i = \frac{\det(A_i)}{\det(A)} \qquad i = 1, \ldots, n$$

Gambar 2.7.1. Solusi Sistem Persamaan Linear Kaidah Cramer

di mana A_i adalah matriks yang dibentuk dengan mengganti kolom ke-i dari A dengan vektor kolom b (Wikipedia, 2017).

2.8 Interpolasi Polinom

Bila data diketahui memiliki ketelitian yang sangat tinggi, maka kurva cocokannya dibuat melalui setiap titik. Kita katakan di sini bahwa kita menginterpolasi titik-titik data dengan sebuah fungsi. Bila fungsi cocokan yang digunakan berbentuk polinom, polinom tersebut dinamakan polinom interpolasi. Pekerjaan menginterpolasi titik data dengan sebuah polinom disebut interpolasi (dengan) polinom (Munir, 2010).

Misalkan terdapat sebuah persoalan yang melibatkan n+1 buah titik berbeda: (x_o, y_o), (x_1,y_1), …, (x_n ,y_n) dan harus ditentukan polinom $p_n(x)$ yang menginterpolasi (melewati) semua titik- tersebut sedemikian rupa sehingga $y = p_n(x_i)$ untuk i = 0, 1, 2, …, n. Nilai y_i dapat berasal dari fungsi f(x) sedemikian sehingga $y_i = f(x_i)$, atau y_i berasal dari nilai empiris yang diperoleh melalui percobaan atau pengamatan. Fungsi $p_n(x)$ ini disebut fungsi hampiran terhadap f(x). Setelah polinom interpolasi $p_n(x)$ ditemukan, $p_n(x)$ dapat digunakan untuk menghitung perkiraan nilai y di x = a, yaitu $y = p_n(a)$. Bergantung pada nilai letaknya, nilai x = a mungkin terletak di dalam rentang titik-titik data ($x_o < a < x_n$) yang merupakan rentang interpolasi atau di luar rentang titik-titik data ($x_o < a < x_n$) yang merupakan rentang rentang ekstrapolasi (Munir, 2010).

2.9 Interpolasi Bicubic Spline

Menurut Syaj (2019), interpolasi adalah suatu proses untuk menentukan nilai baru di suatu posisi yang terletak diantara beberapa sampel. Penentuan nilai baru tersebut dilakukan dengan suatu fungsi tertentu . Interpolasi citra digital bekerja secara dua arah. Proses ini berusaha untuk mendapatkan perkiraan nilai piksel warna dan intensitas yang terbaik berdasarkan nilai pada piksel-piksel di sekitarnya.

Interpolasi memiliki jenis yang sering digunakan dalam penelitian meliputi: tetangga terdekat, *bilinear*, *bicubic*, *spline*, *sinc*, *lanczos* dan lain-lain. Semakin banyak piksel yang berdekatan maka akan lebih akurat, tapi ini memerlukan waktu pemrosesan yang lebih lama lagi. Algoritma ini dapat digunakan untuk mendistorsi dan merubah ukuran citra. Berikut rumus untuk interpolasi:

$$f(x,\ y) = \sum_{i,j=0}^{N=1} a_{i,j}\, x^{i}\, y^{j}$$

a: lokasi piksel tetangga terdekat

b: lokasi piksel baru horisontal

c: lokasi piksel baru vertikal.

Interpolasi *bicubic* adalah interpolasi dengan metode yang lebih canggih dan hasilnya lebih halus pada bagian tepi tepinya daripada interpolasi *bilinear*. Interpolasi *bicubic* menggunakan 4x4 piksel tetangga untuk mendapatkan informasi. Interpolasi *bicubic* menghasilkan gambar yang terasa lebih tajam dibandingkan metode bilinear dan metode *nearest neighbor*.

2.10 Regresi Linear

Menurut Taylor (2024), regresi linear berganda mengacu pada teknik statistik yang digunakan untuk memprediksi hasil suatu variabel berdasarkan nilai dari dua atau lebih variabel lainnya. Teknik ini kadang-kadang disebut sebagai regresi berganda dan merupakan perluasan dari regresi linear. Variabel yang ingin kita prediksi disebut variabel dependen, sedangkan variabel-variabel yang digunakan untuk memprediksi nilai dari variabel dependen disebut variabel independen atau variabel penjelas.

Rumus regresi linear berganda adalah:

$$yi \;=\; \beta0 \;+\; \beta1xi1 \;+\; \beta2xi2 \;+\; \ldots \;+\; \beta pxip \;+\; \epsilon$$

yi adalah variabel dependen atau variabel yang diprediksi.
β0 adalah intersep (nilai y saat x1x1 dan x2x2 sama dengan 0).
β1 dan β2 adalah koefisien regresi yang menunjukkan perubahan pada y terhadap perubahan satu satuan pada masing-masing x1 dan x2.
βp adalah koefisien kemiringan untuk setiap variabel independen.
ϵ adalah kesalahan residua**l** atau galat acak model.

Regresi linear sederhana memungkinkan ahli statistik untuk memprediksi nilai satu variabel menggunakan informasi yang tersedia tentang variabel lain. Regresi linear mencoba membangun hubungan antara dua variabel sepanjang garis lurus, sedangkan regresi berganda adalah jenis regresi di mana variabel dependen menunjukkan hubungan linear dengan dua atau lebih variabel independen. Ini juga dapat bersifat non-linear, di mana variabel dependen dan independen tidak mengikuti garis lurus.

2.11 Regresi Kuadratik Berganda

Menurut Sinha (2013), regresi polinomial dapat diterapkan pada satu variabel prediktor yang disebut regresi polinomial sederhana, atau dapat dihitung pada beberapa variabel prediktor sebagai regresi polinomial berganda. Regresi polinomial berganda orde kedua dapat dinyatakan sebagai:

$$y \;=\; \beta o \;+\; \beta1x1 \;+\; \beta2x2 \;+\; \beta11x1^2 \;+\; \beta22x2^2 \;+\; \beta12x1x2$$

Di sini, β1, β2 disebut sebagai parameter linear. β11, β22 disebut sebagai parameter efek kuadratik. β12 disebut sebagai parameter efek interaksi.
Fungsi regresinya dinyatakan sebagai:

$$E(y) \;=\; \beta o \;+\; \beta1x1 \;+\; \beta2x2 \;+\; \beta11x1^2 \;+\; \beta22x2^2 \;+\; \beta12x1x2$$

Ini juga disebut sebagai Permukaan Respon (Response Surface). Persamaan ini juga dapat diwakili dalam bentuk matriks.

$$Y \;=\; \beta x \;+\; \epsilon$$

Anda dapat melakukan regresi kuadratik multivariat dengan cara yang umum. Mari beri label indeks baris (dan kolom) dari matriks desain A, serta indeks baris dari vektor nilai b, dengan indeks $s(\{p1,p2,p3,\cdots\})$ yang berkaitan dengan koefisien dari $xp1ixp22\cdots$. Sebagai contoh, baris yang diberi label $s(\{1,0,2\})$ akan menjadi baris yang terkait dengan koefisien dari $x1x23$.

Sebagai contoh, untuk regresi kuadratik dua variabel

$y = a + bu + cv + d^2 + euv + fv^2$, Anda perlu menyelesaikan:

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

Gambar 2.11.1. Matriks Pendekatan Pertama Regresi Kuadratik Berganda

dengan a*,b*,c*,d*,e*,f*a*,b*,c*,d*,e*,f* adalah nilai optimal dari a,b,c,d,e,fa,b,c,d,e,f setelah dilakukan regresi kuadratik.

Sebagai alternatif, kita dapat mempertimbangkan:

$$\mathbf{Y} = \mathbf{X} \cdot \begin{pmatrix} a \\ \dots \\ f \end{pmatrix}$$

$$\underbrace{\begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_n \end{pmatrix}}_{\mathbf{Y}} = \underbrace{\begin{pmatrix} 1 & u_1 & v_1 & u_1^2 & u_1v_1 & v_1^2 \\ 1 & u_2 & v_2 & u_2^2 & u_2v_2 & v_2^2 \\ 1 & u_3 & v_3 & u_3^2 & u_3v_3 & v_3^2 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & u_n & v_n & u_n^2 & u_nv_n & v_n^2 \end{pmatrix}}_{\mathbf{X}} \cdot \begin{pmatrix} a \\ b \\ c \\ d \\ e \\ f \end{pmatrix}$$

Gambar 2.11.2. Matriks Pendekatan Kedua Regresi Kuadratik Berganda

Kita bisa menggunakan rumus *Ordinary Least Squares* (OLS) dan mendapatkan solusi:

$$\begin{pmatrix} a^* \\ b^* \\ c^* \\ d^* \\ e^* \\ f^* \end{pmatrix} = (\underbrace{\mathbf{X}^\top \cdot \mathbf{X}}_{\mathbf{A}})^{-1} \cdot \underbrace{\mathbf{X}^\top \cdot \vec{y}}_{\mathbf{b}}$$

Gambar 2.11.3. Rumusan *Ordinary Least Squares*

BAB III

IMPLEMENTASI PUSTAKA DAN PROGRAM

3.1 ADT Matrix

Berisi ADT Matrix serta konstruktor, selektor, atribut, dan *method* lainnya yang digunakan dalam operasi matriks secara umum. Terdapat dalam file Matrix.java yang memiliki *class* Matrix.

Tabel 3.1.1. Konstruktor ADT Matrix

Konstruktor	Deskripsi
public Matrix(int rows, int cols)	Membuat matriks dengan jumlah baris sebanyak *rows* dan jumlah kolom sebanyak *cols*

Tabel 3.1.2. Selektor ADT Matrix

Selektor	Deskripsi
public Matrix(int rows, int cols)	Membuat matriks dengan jumlah baris sebanyak *rows* dan jumlah kolom sebanyak *cols*
public double get(int row, int col)	Mengembalikan elemen dari matriks pada baris *row* dan kolom *col*
public void set(int row, int col, double value)	Mengubah elemen dari matriks pada baris *row* dan kolom *col*
public int get_rows()	Mengembalikan jumlah baris pada matriks
public int get_cols()	Mengembalikan jumlah kolom pada matriks

Tabel 3.1.3. *Method* ADT Matrix

Method	Deskripsi
public Matrix copy_matrix()	Mengembalikan salinan dari matriks
public double[] copy_row(int row)	Mengembalikan salinan dari baris *row* pada matriks

public double[] copy_col(int col)	Mengembalikan salinan dari kolom *col* pada matriks
public Matrix multiply_by_constant(double constant)	Mengembalikan salinan dari matriks yang setiap elemennya telah dikalikan dengan konstanta *constant*
public void print_matrix()	Menampilkan seluruh elemen matriks ke layar
public Matrix transpose_matrix()	Mengembalikan salinan dari matriks yang telah ditukar elemen baris dengan elemen kolomnya (di-transpose)
public Matrix multiply(Matrix other)	Mengembalikan hasil perkalian matriks dengan matriks lain, yakni Matrix *other*
public static void save_output_SPL(Scanner scanner, Matrix result)	Menyimpan hasil solusi SPL dari Matrix *matrix* ke dalam file .txt

3.2 Eliminasi Gauss

Berisi *method* untuk melakukan eliminasi Gauss terhadap matriks sehingga dihasilkan matriks eselon baris. Terdapat dalam file GaussianElimination.java yang memiliki *class* GaussianElimination.

Tabel 3.2.1. *Method* Eliminasi Gauss

Method	Deskripsi
private static void swap_rows(Matrix matrix, int row1, int row2)	Melakukan pertukaran antara baris *row1* dengan baris *row2* pada Matrix *matrix*
public static Matrix gaussian_elimination(Matrix matrix)	Mengembalikan salinan dari Matrix *matrix* yang telah diproses dengan operasi baris elementer (OBE) sehingga berbentuk matriks eselon baris
private static int find_non_zero(Matrix matrix, int row)	Mengembalikan indeks pertama yang bernilai tidak 0 pada matriks di baris yang diberikan. Jika tidak ditemukan, fungsi mengembalikan -1.

3.3 Eliminasi Gauss-Jordan

Berisi *method* untuk melakukan eliminasi Gauss-Jordan terhadap matriks sehingga dihasilkan matriks eselon baris tereduksi. Terdapat dalam file GaussJordanElimination.java yang memiliki *class* GaussJordanElimination.

Tabel 3.3.1. *Method* Eliminasi Gauss-Jordan

Method	Deskripsi
private static int find_non_zero(Matrix matrix, int row)	Mengembalikan indeks pertama yang bernilai tidak 0 pada matriks di baris yang diberikan. Jika tidak ditemukan, fungsi mengembalikan -1.
public static Matrix gauss_jordan_elimination(Matrix matrix)	Mengembalikan salinan dari Matrix *matrix* yang telah diproses dengan operasi baris elementer (OBE) sehingga berbentuk matriks eselon baris tereduksi

3.4 Determinan

Berisi *method* untuk mencari determinan dari matriks, baik dengan ekspansi kofaktor dalam file DeterminantCofactor.java yang memiliki *class* DeterminantCofactor maupun dengan reduksi baris dalam file DeterminantRowReduction.java yang memiliki *class* DeterminantRowReduction.

Tabel 3.4.1. *Method* Determinan dengan Ekspansi Kofaktor

Method	Deskripsi
private static Matrix get_minor(Matrix matrix, int row, int col)	Mengembalikan minor dari elemen Matrix *matrix* pada baris *row* dan kolom *col*
public static double cofactor(Matrix matrix, int row, int col)	Mengembalikan kofaktor dari elemen Matrix *matrix* pada baris *row* dan kolom *col*
private static double determinant_cofactor(Matrix matrix)	Mengembalikan determinan dari Matrix *matrix* yang dihitung menggunakan metode ekspansi kofaktor
public static double determinant(Matrix matrix)	Mengembalikan determinan dari Matrix *matrix* dengan memvalidasi dimensi Matrix *matrix* dan memanggil method determinan_cofactor(matrix)

Tabel 3.4.2. *Method* Determinan dengan Reduksi Baris

Method	Deskripsi
private static void swap_rows(Matrix matrix, int row1, int row2)	Melakukan pertukaran antara baris *row1* dengan baris *row2* pada Matrix *matrix*
public static double determinant_row_reduction(Matrix matrix)	Mengembalikan determinan dari Matrix *matrix* yang dihitung menggunakan metode reduksi baris

3.5 Matriks Balikan

Berisi *method* untuk mencari matriks balikan atau invers, baik dengan eliminasi Gauss-Jordan atau operasi baris elementer (OBE) dalam file InverseMatrixGaussJordan.java yang memiliki *class* InverseMatrixGaussJordan maupun dengan adjoin dan determinan dalam file InverseMatrixDeterminant.java yang memiliki *class* InverseMatrixDeterminant.

Tabel 3.5.1. *Method* Matriks Balikan dengan Operasi Baris Elementer (OBE)

Method	Deskripsi
public static Matrix inverse_matrix(Matrix matrix)	Mengembalikan matriks balikan dari Matrix *matrix* yang dihitung menggunakan operasi baris elementer (OBE)

Tabel 3.5.2. *Method* Matriks Balikan dengan Adjoin dan Determinan

Method	Deskripsi
public static Matrix inverse_with_determinant(Matrix matrix)	Mengembalikan matriks balikan dari Matrix *matrix* yang dihitung menggunakan adjoin dan determinan
private static Matrix inverse_2x2 (Matrix matrix)	Mengembalikan matriks balikan dari Matrix *matrix* yang berdimensi 2x2

3.6 Sistem Persamaan Linear (SPL)

Berisi *method* untuk mencari solusi dari sistem persamaan linear (SPL) yang dapat memiliki solusi unik, solusi banyak (parametrik), atau tidak ada solusi. SPL diselesaikan dengan empat cara, yaitu dengan eliminasi Gauss dalam file SPLGaussian.java yang memiliki *class* SPLGaussian, dengan eliminasi Gauss-Jordan dalam file SPLGaussJordan.java yang memiliki *class* SPLGaussJordan, dengan matriks balikan dalam file SPLInverse.java yang memiliki *class* SPLInverse, dan dengan aturan Cramer dalam file SPLCramer.java yang memiliki *class* SPL

Tabel 3.6.1. *Method* SPL dengan Eliminasi Gauss

Method	Deskripsi
public static void spl_gaussian(Matrix matrix, Scanner scanner)	Menghitung solusi dari SPL berdasarkan Matrix *matrix* dengan metode eliminasi Gauss

Tabel 3.6.2. *Method* SPL dengan Eliminasi Gauss-Jordan

Method	Deskripsi
public static void spl_gauss_jordan(Matrix matrix, Scanner scanner)	Menghitung solusi dari SPL berdasarkan Matrix *matrix* dengan metode eliminasi Gauss-Jordan

Tabel 3.6.3. *Method* SPL dengan Matriks Balikan

Method	Deskripsi
private static void swap_rows(Matrix matrix, int row1, int row2)	Melakukan pertukaran antara baris *row1* dengan baris *row2* pada Matrix *matrix*
public static Matrix basis(Matrix matrix)	Mengembalikan basis dari Matrix *matrix* untuk membantu perhitungan matriks balikan karena hanya matriks yang baris-barisnya saling independen (basis) dapat memiliki invers
public static void spl_inverse(Matrix matrix, Scanner scanner)	Menghitung solusi dari SPL berdasarkan Matrix *matrix* dengan menggunakan matriks balikan

Tabel 3.6.4. *Method* SPL dengan Aturan Cramer

Method	Deskripsi
private static void swap_rows(Matrix matrix, int row1, int row2)	Melakukan pertukaran antara baris *row1* dengan baris *row2* pada Matrix *matrix*
public static Matrix basis(Matrix matrix)	Mengembalikan basis dari Matrix *matrix*

	untuk membantu perhitungan matriks balikan karena hanya matriks yang baris-barisnya saling independen (basis) dapat memiliki invers
private static double determinant_n(Matrix matrix, int n)	Mengembalikan determinan dari Matrix *matrix* yang kolom ke-*n* nya sudah diganti dengan kolom konstanta dari *matrix* dan tidak lagi berbentuk matriks *augmented*
public static void spl_cramer(Matrix matrix, Scanner scanner)	Menghitung solusi dari SPL berdasarkan Matrix *matrix* dengan menggunakan aturan Cramer

3.7 Interpolasi Polinomial

Berisi *method* untuk melakukan interpolasi polinomial, yakni mencari nilai y pada nilai x yang terdapat dalam rentang nilai x yang dimasukkan. Interpolasi polinomial dilakukan berdasarkan titik (x dan y) yang dimasukkan. Terdapat dalam file PolynomialInterpolation.java yang memiliki *class* PolynomialInterpolation.

Tabel 3.7.1. *Method* Interpolasi Polinomial

Method	Deskripsi
public static Matrix polynomial_interpolation(Matrix points, int idx)	Mengembalikan koefisien interpolasi polinomial dari Matrix *points* yang berisi titik-titik x dan y dengan proses pencarian koefisien memanfaatkan salah satu dari *method* SPL sesuai pilihan *idx*
public static void estimate_value(Matrix coefficients, double x)	Menghitung nilai y pada nilai *x* yang diminta dengan memanfaatkan koefisien *coefficients* interpolasi polinomial dari matriks
public static Matrix spl_gaussian(Matrix matrix)	Mengembalikan salinan dari Matrix *matrix* yang telah diproses dengan operasi baris elementer (OBE) sehingga berbentuk matriks eselon baris
public static Matrix spl_gauss_jordan(Matrix matrix)	Mengembalikan salinan dari Matrix *matrix* yang telah diproses dengan operasi baris elementer (OBE) sehingga berbentuk matriks eselon baris tereduksi
private static Matrix gauss_solution(Matrix	Mengembalikan matriks berisi solusi dari SPL

matrix)	berdasarkan Matrix *matrix* yang berbentuk matriks eselon baris atau matriks eselon baris tereduksi
public static Matrix spl_inverse(Matrix matrix)	Mengembalikan matriks berisi solusi dari SPL berdasarkan Matrix *matrix* dengan metode matriks balikan
public static Matrix spl_cramer(Matrix matrix)	Mengembalikan matriks berisi solusi dari SPL berdasarkan Matrix *matrix* dengan aturan Cramer
private static double determinant_n(Matrix matrix, int n)	Mengembalikan determinan dari Matrix *matrix* yang kolom ke-*n* nya sudah diganti dengan kolom konstanta dari *matrix* dan tidak lagi berbentuk matriks *augmented* untuk proses SPL dengan aturan Cramer

3.8 *Bicubic Spline Interpolation*

Berisi *method* untuk menghitung interpolasi suatu titik berdasarkan titik-titik lain di sekitarnya menggunakan metode *bicubic spline interpolation*.

Tabel 3.8.1. *Method Bicubic Spline Interpolation*

Method	Deskripsi
private static double power(double b, double p)	Menghitung nilai dari a^b, dengan 0^0 dianggap 1.
private static double Mvalue(int type, double x, double y, int i, int j)	Menghitung elemen baris i kolom j pada matriks transformasi dengan x dan y sebagai parameter tambahan.
private static Matrix bicubic_spline_interpolation(Matrix F)	Menghitung hasil kali antara matriks transformasi dengan F sebagai matriks yang akan ditransformasikan.
public static double evaluate(Matrix F, double x, double y)	Menghitung nilai hasil interpolasi pada titik (x, y) terhadap matriks F.

3.9 Regresi Linear Berganda

Berisi *method* untuk mencari nilai *bias* dan *weight* dari masing-masing fitur *training data* menggunakan metode *ordinary least squares* (OLS) atau Ridge *regression* serta melakukan *prediction* berdasarkan *input* xk.

Tabel 3.9.1. *Method* Regresi Linear Berganda

Method	Deskripsi
public static void multiple_linear_regression(Matrix matrix, Matrix xk)	Menghasilkan *output weight* dan *bias* berdasarkan data *training* dan juga hasil *prediction* berdasarkan *input* xk.
public static void matrix_regression_output_file(Scanner scanner, Matrix weights, Matrix prediction)	Melakukan validasi pada *user* untuk menghasilkan output pada sebuah *file* dengan nama sesuai input *user* dan mengikuti format yang telah ditetapkan (*weights* dan hasil *prediction)*.

3.10 Regresi Kuadratik Berganda

Berisi *method* untuk mencari nilai *bias* dan *weight* dari masing-masing fitur *training data* yang telah diekspansi hingga derajat polinom kedua menggunakan metode *ordinary least squares* (OLS) atau Ridge *regression* serta melakukan *prediction* berdasarkan *input* xk.

Tabel 3.10.1. *Method* Regresi Kuadratik Berganda

Method	Deskripsi
public static void multiple_quadratic_regression(Matrix matrix, Matrix xk)	Menghasilkan *output weight* dan *bias* berdasarkan data *training* yang telah diekspansi hingga derajat polinom kedua dan juga hasil *prediction* berdasarkan *input* xk.
public static void matrix_regression_output_file(Scanner scanner, Matrix weights, Matrix prediction, String[] variables)	Melakukan validasi pada *user* untuk menghasilkan output pada sebuah *file* dengan nama sesuai input *user* dan mengikuti format yang telah ditetapkan (*weights* dan hasil *prediction*).

3.11 *Image Resizing* and *Stretching*

Berisi *method* untuk melakukan *image resizing,* yaitu *method* untuk memperbesar atau memperkecil *image* berdasarkan skala yang diberikan: skala horizontal dan vertikal dapat berbeda.

Tabel 3.11.1. *Method* Interpolasi Gambar

Method	Deskripsi
private static Matrix discrete_derivative(Matrix neighbors)	Menghitung turunan diskrit terhadap x, y, dan xy pada titik-titik (1, 1), (1, 2), (2, 1), (2, 2) relatif terhadap matriks *neighbor* (0-*based indexing*).
private static int bicubic_spline_interpolation(BufferedImage image, double x, double y)	Menghitung interpolasi nilai alpha, *red*, *green,* dan *blue* pada posisi (x, y) kemudian digabungkan untuk membentuk nilai pixel yang baru.
public static BufferedImage resize(BufferedImage source, double scaleX, double scaleY)	Memperkecil atau memperbesar *image source* sesuai skala yang diberikan. Skala perubahan horizontal sebesar scaleX dan skala perubahan vertikal sebesar scaleY.
public static void bicubic_spline_output_file(Scanner scanner, double[] results, double[] x_comps, double[] y_comps, double results_size)	Melakukan validasi pada *user* untuk menghasilkan output pada sebuah *file* dengan nama sesuai input *user* dan mengikuti format yang telah ditetapkan (hasil *prediction*).

3.12 Program Utama

Berisi *method* untuk menjalankan program Sistem Persamaan Linier, Determinan, dan Aplikasinya secara keseluruhan. Terdapat dalam file Main.java yang memiliki *class* Main.

Tabel 3.12.1. *Method* Program Utama

Method	Deskripsi
public static void main(String[] args)	Menjalankan program secara keseluruhan dengan menampilkan 8 menu utama, yaitu sistem persamaan linear, determinan, matriks balikan, interpolasi polinomial, interpolasi *bicubic spline*, regresi linear dan kuadratik berganda, interpolasi gambar (*scaling*), dan keluar, serta memanggil fitur-fitur yang telah dibuat
public static Matrix input_matrix(int rows, int cols, Scanner scanner)	Mengembalikan matriks berdimensi baris *rows* dan kolom *cols* berdasarkan hasil masukan dari keyboard pengguna dengan memanfaatkan *scanner*

public static Matrix augment_matrix(Matrix a, Matrix b)	Mengembalikan matriks *augmented* dari Matrix *a* dan Matrix *b* dengan susunan [*a* | *b*]
public static Matrix input_matrix_file(Scanner scanner)	Mengembalikan matriks berdasarkan masukan dari file pengguna dengan memanfaatkan *scanner*
public static double[][] input_matrix_file_interpolation(double[] x, Integer[] len, Scanner scanner)	Mengembalikan matriks dengan format khusus untuk interpolasi berdasarkan masukan dari file pengguna dengan memanfaatkan *scanner*
public static Matrix[] input_matrix_file_regression(Scanner scanner)	Membaca input dari *file* menggunakan format yang telah disesuaikan (n, m, x, y, xk) untuk menghitung *bias* dan *weight* regression beserta *prediction* xk menggunakan metode regresi linear berganda atau regresi kuadratik berganda.
public static Matrix[] input_matrix_file_bicubic(Scanner scanner)	Membaca input dari *file* menggunakan format yang sudah disesuaikan untuk menghitung interpolasi menggunakan metode *bicubic spline interpolation*.

BAB IV
EKSPERIMEN

4.1 Studi dan Tes Kasus Solusi SPL Ax = b

Berikut adalah hasil eksperimen studi dan tes kasus solusi SPL Ax = b:

Tabel 4.1.1. Hasil Pengujian Studi dan Tes Kasus SPL Ax = b

No.	Soal dan Hasil Eksperimen
a.	

1) Soal

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

2) Hasil Eksperimen

a) Input *User*

```
MENU
1. Sistem Persamaan Linier
2. Determinan
3. Matriks Balikan
4. Interpolasi Polinom
5. Interpolasi Bicubic Spline
6. Regresi Linier dan Kuadratik Berganda
7. Scaling Gambar
8. Keluar
Menu yang ingin diakses: 1
1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Matriks balikan
4. Kaidah Cramer
Sub-menu yang ingin diakses: 1
1. Input dari keyboard
2. Input dari file
Cara input matrix:
```

b) SPL Eliminasi Gauss

```
1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Matriks balikan
4. Kaidah Cramer
Sub-menu yang ingin diakses: 1
1. Input dari keyboard
2. Input dari file
Cara input matrix: 2
Masukkan nama file:
../test/testcase1a.txt
Matrix does not have a solution.
```

c) SPL Eliminasi Gauss-Jordan

```
1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Matriks balikan
4. Kaidah Cramer
Sub-menu yang ingin diakses: 2
1. Input dari keyboard
2. Input dari file
Cara input matrix: 2
Masukkan nama file:
../test/testcase1a.txt
Matrix does not have a solution.
```

d) SPL Matriks Balikan

```
1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Matriks balikan
4. Kaidah Cramer
Sub-menu yang ingin diakses: 3
1. Input dari keyboard
2. Input dari file
Cara input matrix: 2
Masukkan nama file:
../test/testcase1a.txt
Matrix does not have a solution.
```

e) SPL Kaidah Cramer

```
1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Matriks balikan
4. Kaidah Cramer
Sub-menu yang ingin diakses: 4
1. Input dari keyboard
2. Input dari file
Cara input matrix: 2
Masukkan nama file:
../test/testcase1a.txt
Matrix does not have a solution.
```

b.

1) Soal

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

2) Hasil Eksperimen

a) SPL Eliminasi Gauss

```
1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Matriks balikan
4. Kaidah Cramer
Sub-menu yang ingin diakses: 1
1. Input dari keyboard
2. Input dari file
Cara input matrix: 2
Masukkan nama file:
../test/testcase1b.txt
Matrix has infinite solutions.
Parametric solution:
x1 = 1.0t3 + 3.0
x2 = 2.0t3 +
x3 = t1
x4 = 1.0t3 + -1.0
x5 = t2
```

b) SPL Eliminasi Gauss-Jordan

```
1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Matriks balikan
4. Kaidah Cramer
Sub-menu yang ingin diakses: 2
1. Input dari keyboard
2. Input dari file
Cara input matrix: 2
Masukkan nama file:
../test/testcase1b.txt
Matrix has infinite solutions.
Parametric solution:
x1 = 1.0t3 + 3.0
x2 = 2.0t3 +
x3 = t1
x4 = 1.0t3 + -1.0
x5 = t2
```

c) SPL Matriks Balikan

```
1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Matriks balikan
4. Kaidah Cramer
Sub-menu yang ingin diakses: 3
1. Input dari keyboard
2. Input dari file
Cara input matrix: 2
Masukkan nama file:
../test/testcase1b.txt
Matrix has infinite solutions.
Parametric solution:
x1 = 1.0t3 + 3.0
x2 = 2.0t3 +
x3 = t1
x4 = 1.0t3 + -1.0
x5 = t2
```

	d) SPL Kaidah Cramer
	`1. Metode eliminasi Gauss`
	`2. Metode eliminasi Gauss-Jordan`
	`3. Matriks balikan`
	`4. Kaidah Cramer`
	`Sub-menu yang ingin diakses: 4`
	`1. Input dari keyboard`
	`2. Input dari file`
	`Cara input matrix: 2`
	`Masukkan nama file:`
	`../test/testcase1b.txt`
	`The Cramer method is used for square matrix, i.e. the number of rows is equal to the number of columns.`

c.	1) Soal
	$$A = \begin{bmatrix} 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}, \quad b = \begin{bmatrix} 2 \\ -1 \\ 1 \end{bmatrix}$$
	2) Hasil Eksperimen
	a) SPL Eliminasi Gauss
	`1. Metode eliminasi Gauss`
	`2. Metode eliminasi Gauss-Jordan`
	`3. Matriks balikan`
	`4. Kaidah Cramer`
	`Sub-menu yang ingin diakses: 1`
	`1. Input dari keyboard`
	`2. Input dari file`
	`Cara input matrix: 2`
	`Masukkan nama file:`
	`../test/testcase1c.txt`
	`Matrix has infinite solutions.`
	`Parametric solution:`
	`x1 = t1`
	`x2 = -1.0t3 + 1.0`
	`x3 = t2`
	`x4 = -1.0t3 + -2.0`
	`x5 = 1.0t3 + 1.0`
	`x6 = t3`
	b) SPL Eliminasi Gauss-Jordan
	`1. Metode eliminasi Gauss`
	`2. Metode eliminasi Gauss-Jordan`
	`3. Matriks balikan`
	`4. Kaidah Cramer`
	`Sub-menu yang ingin diakses: 2`
	`1. Input dari keyboard`
	`2. Input dari file`
	`Cara input matrix: 2`
	`Masukkan nama file:`
	`../test/testcase1c.txt`
	`Matrix has infinite solutions.`
	`Parametric solution:`
	`x1 = t1`
	`x2 = -1.0t3 + 1.0`
	`x3 = t2`
	`x4 = -1.0t3 + -2.0`
	`x5 = 1.0t3 + 1.0`
	`x6 = t3`

c) SPL Matriks Balikan

```
1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Matriks balikan
4. Kaidah Cramer
Sub-menu yang ingin diakses: 3
1. Input dari keyboard
2. Input dari file
Cara input matrix: 2
Masukkan nama file:
../test/testcase1c.txt
Matrix has infinite solutions.
Parametric solution:
x1 = t1
x2 = -1.0t3 + 1.0
x3 = t2
x4 = -1.0t3 + -2.0
x5 = 1.0t3 + 1.0
x6 = t3
```

d) SPL Kaidah Cramer

```
1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Matriks balikan
4. Kaidah Cramer
Sub-menu yang ingin diakses: 4
1. Input dari keyboard
2. Input dari file
Cara input matrix: 2
Masukkan nama file:
../test/testcase1c.txt
The Cramer method is used for square matrix, i.e. the number of rows is equal to the number of columns.
```

d.

1) Soal

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

2) Hasil Eksperimen n = 6

a) SPL Eliminasi Gauss

```
1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Matriks balikan
4. Kaidah Cramer
Sub-menu yang ingin diakses: 1
1. Input dari keyboard
2. Input dari file
Cara input matrix: 2
Masukkan nama file:
../test/testcase1d1.txt
x1 = 36.00000000098032, x2 = -630.0000000292666, x3 = 3360.000000203484, x4 = -7560.000000539232, x5 = 7560.000000603351
, x6 = -2772.000000240222
```

b) SPL Eliminasi Gauss-Jordan

```
1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Matriks balikan
4. Kaidah Cramer
Sub-menu yang ingin diakses: 2
1. Input dari keyboard
2. Input dari file
Cara input matrix: 2
Masukkan nama file:
../test/testcase1d1.txt
x1 = 36.000000000980656, x2 = -630.0000000292673, x3 = 3360.000000203484, x4 = -7560.000000539232, x5 = 7560.00000060335
1, x6 = -2772.000000240222
```

c) SPL Matriks Balikan

```
1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Matriks balikan
4. Kaidah Cramer
Sub-menu yang ingin diakses: 3
1. Input dari keyboard
2. Input dari file
Cara input matrix: 2
Masukkan nama file:
../test/testcase1d1.txt
x1 = 36.000000000980656, x2 = -630.0000000292673, x3 = 3360.000000203484, x4 = -7560.000000539232, x5 = 7560.00000060335
1, x6 = -2772.000000240222
```

d) SPL Kaidah Cramer

```
1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Matriks balikan
4. Kaidah Cramer
Sub-menu yang ingin diakses: 4
1. Input dari keyboard
2. Input dari file
Cara input matrix: 2
Masukkan nama file:
../test/testcase1d1.txt
x1 = 36.0000000013453, x2 = -630.0000000277283, x3 = 3360.00000020911, x4 = -7560.000000547803, x5 = 7560.000000613385,
x6 = -2772.0000002445704
```

3) Hasil Eksperimen n = 10

a) SPL Eliminasi Gauss

```
1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Matriks balikan
4. Kaidah Cramer
Sub-menu yang ingin diakses: 1
1. Input dari keyboard
2. Input dari file
Cara input matrix: 2
Masukkan nama file:
../test/testcase1d2.txt
x1 = 99.99634656422131, x2 = -4949.682659989863, x3 = 79193.21483185585, x4 = -600538.1369470609, x5 = 2522224.258680413
5, x6 = -6305485.547186596, x7 = 9608261.783228857, x8 = -8750305.214203862, x9 = 4375119.565502384, x10 = -923630.23495
73893
```

b) SPL Eliminasi Gauss-Jordan

```
1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Matriks balikan
4. Kaidah Cramer
Sub-menu yang ingin diakses: 2
1. Input dari keyboard
2. Input dari file
Cara input matrix: 2
Masukkan nama file:
../test/testcase1d2.txt
x1 = 99.99634656426257, x2 = -4949.682659994306, x3 = 79193.21483186251, x4 = -600538.1369470842, x5 = 2522224.25868042,
 x6 = -6305485.547186624, x7 = 9608261.78322886, x8 = -8750305.214203862, x9 = 4375119.565502384, x10 = -923630.23495738
93
```

<table>
<tr>
<td></td>
<td>

c) SPL Matriks Balikan

<pre>
1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Matriks balikan
4. Kaidah Cramer
Sub-menu yang ingin diakses: 3
1. Input dari keyboard
2. Input dari file
Cara input matrix: 2
Masukkan nama file:
../test/testcase1d2.txt
x1 = 99.99634656426257, x2 = -4949.682659994306, x3 = 79193.21483186251, x4 = -600538.1369470842, x5 = 2522224.25868042,
 x6 = -6305485.547186624, x7 = 9608261.78322886, x8 = -8750305.214203862, x9 = 4375119.565502384, x10 = -923630.23495738
93
</pre>

d) SPL Kaidah Cramer

<pre>
1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Matriks balikan
4. Kaidah Cramer
Sub-menu yang ingin diakses: 4
1. Input dari keyboard
2. Input dari file
Cara input matrix: 2
Masukkan nama file:
../test/testcase1d2.txt
x1 = 99.9924654760246, x2 = -4949.557030480041, x3 = 79193.25392874563, x4 = -600537.4838678582, x5 = 2522224.827346026,
 x6 = -6305486.515133805, x7 = 9608263.464827241, x8 = -8750306.935237568, x9 = 4375120.511596158, x10 = -923630.4510748
411
</pre>

</td>
</tr>
</table>

4.2 Studi dan Tes Kasus Solusi SPL Berbentuk Matriks *Augmented*

Berikut adalah hasil eksperimen studi dan tes kasus solusi SPL berbentuk matriks *augmented*:

Tabel 4.2.1. Hasil Pengujian Studi dan Tes Kasus SPL Berbentuk Matriks *Augmented*

<table>
<tr>
<th>No.</th>
<th>Soal dan Hasil Eksperimen</th>
</tr>
<tr>
<td>a.</td>
<td>

1) Soal

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

</td>
</tr>
</table>

2) Hasil Eksperimen

a) SPL Eliminasi Gauss

```
1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Matriks balikan
4. Kaidah Cramer
Sub-menu yang ingin diakses: 1
1. Input dari keyboard
2. Input dari file
Cara input matrix: 2
Masukkan nama file:
../test/testcase2a.txt
Matrix has infinite solutions.
Parametric solution:
x1 = 1.0t2 + -1.0
x2 = 2.0t2 +
x3 = t1
x4 = t2
```

b) SPL Eliminasi Gauss-Jordan

```
1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Matriks balikan
4. Kaidah Cramer
Sub-menu yang ingin diakses: 2
1. Input dari keyboard
2. Input dari file
Cara input matrix: 2
Masukkan nama file:
../test/testcase2a.txt
Matrix has infinite solutions.
Parametric solution:
x1 = 1.0t2 + -1.0
x2 = 2.0t2 +
x3 = t1
x4 = t2
```

c) SPL Matriks Balikan

```
1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Matriks balikan
4. Kaidah Cramer
Sub-menu yang ingin diakses: 3
1. Input dari keyboard
2. Input dari file
Cara input matrix: 2
Masukkan nama file:
../test/testcase2a.txt
Matrix has infinite solutions.
Parametric solution:
x1 = 1.0t2 + -1.0
x2 = 2.0t2 +
x3 = t1
x4 = t2
```

d) SPL Kaidah Cramer

```
1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Matriks balikan
4. Kaidah Cramer
Sub-menu yang ingin diakses: 4
1. Input dari keyboard
2. Input dari file
Cara input matrix: 2
Masukkan nama file:
../test/testcase2a.txt
The Cramer method is used for square matrix, i.e. the number of rows is equal to the number of columns.
```

b.

1) Soal

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

2) Hasil Eksperimen

a) SPL Eliminasi Gauss

```
1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Matriks balikan
4. Kaidah Cramer
Sub-menu yang ingin diakses: 1
1. Input dari keyboard
2. Input dari file
Cara input matrix: 2
Masukkan nama file:
../test/testcase2b.txt
x1 = 0.0, x2 = 2.0, x3 = 1.0, x4 = 1.0
Apakah output ingin disimpan ke file? (y/n):
```

b) SPL Eliminasi Gauss-Jordan

```
1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Matriks balikan
4. Kaidah Cramer
Sub-menu yang ingin diakses: 2
1. Input dari keyboard
2. Input dari file
Cara input matrix: 2
Masukkan nama file:
../test/testcase2b.txt
x1 = 0.0, x2 = 2.0, x3 = 1.0, x4 = 1.0
Apakah output ingin disimpan ke file? (y/n):
```

c) SPL Matriks Balikan

```
1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Matriks balikan
4. Kaidah Cramer
Sub-menu yang ingin diakses: 3
1. Input dari keyboard
2. Input dari file
Cara input matrix: 2
Masukkan nama file:
../test/testcase2b.txt
x1 = 0.0, x2 = 2.0, x3 = 1.0, x4 = 0.9999999999999999
Apakah output ingin disimpan ke file? (y/n): n
```

d) SPL Kaidah Cramer

```
1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Matriks balikan
4. Kaidah Cramer
Sub-menu yang ingin diakses: 4
1. Input dari keyboard
2. Input dari file
Cara input matrix: 2
Masukkan nama file:
../test/testcase2b.txt
The Cramer method is used for square matrix, i.e. the number of rows is equal to the number of columns.
```

4.3 Studi dan Tes Kasus Solusi SPL Berbentuk Persamaan

Berikut adalah hasil eksperimen studi dan tes kasus solusi SPL berbentuk persamaan:

Tabel 4.3.1. Hasil Pengujian Studi dan Tes Kasus Solusi SPL Berbentuk Persamaan

No.	Soal dan Hasil Eksperimen
a.	1) Soal

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

2) Hasil Eksperimen

a) SPL Eliminasi Gauss

```
1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Matriks balikan
4. Kaidah Cramer
Sub-menu yang ingin diakses: 1
1. Input dari keyboard
2. Input dari file
Cara input matrix: 2
Masukkan nama file:
../test/testcase3a.txt
x1 = -0.2243243243243243, x2 = 0.18243243243243246, x3 = 0.7094594594594594, x4 = -0.25810810810810797
Apakah output ingin disimpan ke file? (y/n): n
```

b) SPL Eliminasi Gauss-Jordan

```
1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Matriks balikan
4. Kaidah Cramer
Sub-menu yang ingin diakses: 2
1. Input dari keyboard
2. Input dari file
Cara input matrix: 2
Masukkan nama file:
../test/testcase3a.txt
x1 = -0.22432432432432436, x2 = 0.18243243243243246, x3 = 0.7094594594594594, x4 = -0.25810810810810797
Apakah output ingin disimpan ke file? (y/n): n
```

c) SPL Matriks Balikan

```
1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Matriks balikan
4. Kaidah Cramer
Sub-menu yang ingin diakses: 3
1. Input dari keyboard
2. Input dari file
Cara input matrix: 2
Masukkan nama file:
../test/testcase3a.txt
x1 = -0.2243243243243243, x2 = 0.1824324324324324 3, x3 = 0.7094594594594593, x4 = -0.25810810810810814
Apakah output ingin disimpan ke file? (y/n): n
```

d) SPL Kaidah Cramer

```
1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Matriks balikan
4. Kaidah Cramer
Sub-menu yang ingin diakses: 4
1. Input dari keyboard
2. Input dari file
Cara input matrix: 2
Masukkan nama file:
../test/testcase3a.txt
x1 = -0.22432432432432414, x2 = 0.1824324324324324, x3 = 0.7094594594594594, x4 = -0.25810810810810797
Apakah output ingin disimpan ke file? (y/n):
```

b.

1) Soal

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

2) Hasil Eksperimen

a) SPL Eliminasi Gauss

```
1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Matriks balikan
4. Kaidah Cramer
Sub-menu yang ingin diakses: 1
1. Input dari keyboard
2. Input dari file
Cara input matrix: 2
Masukkan nama file:
../test/testcase3b.txt
x1 = 24.004066752506333, x2 = 2532.9888471028607, x3 = -2548.992913855367, x4 = -2548.18835476700
9, x5 = 2816.0, x6 = -252.81164523299108, x7 = -256.0, x8 = 285.0, x9 = -16.0
Apakah output ingin disimpan ke file? (y/n): n
```

b) SPL Eliminasi Gauss-Jordan

```
1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Matriks balikan
4. Kaidah Cramer
Sub-menu yang ingin diakses: 2
1. Input dari keyboard
2. Input dari file
Cara input matrix: 2
Masukkan nama file:
../test/testcase3b.txt
x1 = 24.00406675250905, x2 = 768.0, x3 = 0.0, x4 = 256.0, x5 = -256.0, x6 = -252.81164523299117,
x7 = -256.0, x8 = 285.0, x9 = -16.0
Apakah output ingin disimpan ke file? (y/n): n
```

c) SPL Matriks Balikan

```
1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Matriks balikan
4. Kaidah Cramer
Sub-menu yang ingin diakses: 3
1. Input dari keyboard
2. Input dari file
Cara input matrix: 2
Masukkan nama file:
../test/testcase3b.txt
x1 = 21.59469175250908, x2 = -240.51999999999998, x3 = 18.0, x4 = 694.04, x5 = -679.04, x6 = -224
.20461673299113, x7 = -227.82999999999998, x8 = 254.42062499999997, x9 = -13.590625
Apakah output ingin disimpan ke file? (y/n): n
```

d) SPL Kaidah Cramer

```
1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Matriks balikan
4. Kaidah Cramer
Sub-menu yang ingin diakses: 4
1. Input dari keyboard
2. Input dari file
Cara input matrix: 2
Masukkan nama file:
../test/testcase3b.txt
The Cramer method is used for square matrix, i.e. the number of rows is equal to the number of columns.
```

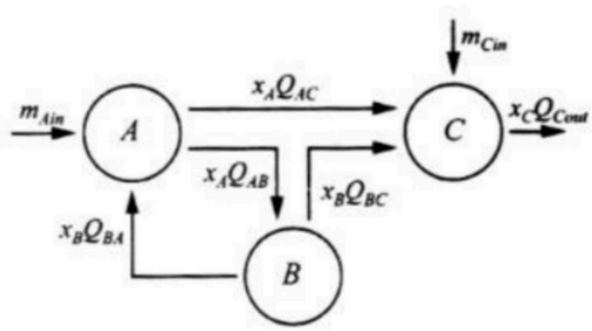
4.4 Studi dan Tes Kasus Sistem Reaktor

Berikut adalah hasil eksperimen studi dan tes kasus sistem reaktor:

Tabel 4.4.1. Hasil Pengujian Studi dan Tes Kasus Sistem Reaktor

<table>
<tr><th>No.</th><th>Soal dan Hasil Eksperimen</th></tr>
<tr><td>a.</td><td>

1) Soal



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

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

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

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

Tentukan solusi x_A, x_B, x_C dengan menggunakan parameter berikut : $Q_{AB} = 40$, $Q_{AC} = 80$, $Q_{BA} = 60$, $Q_{BC} = 20$ dan $Q_{Cout} = 150$ m^3/s dan $m_{Ain} = 1300$ dan $m_{Cin} = 200$ *mg/s*.

2) Hasil Eksperimen

a) SPL Eliminasi Gauss

```
1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Matriks balikan
4. Kaidah Cramer
Sub-menu yang ingin diakses: 1
1. Input dari keyboard
2. Input dari file
Cara input matrix: 2
Masukkan nama file:
../test/testcase4.txt
x1 = 14.444444444444446, x2 = 7.222222222222223, x3 = 10.0
```

b) SPL Eliminasi Gauss-Jordan

</td></tr>
</table>

```
1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Matriks balikan
4. Kaidah Cramer
Sub-menu yang ingin diakses: 2
1. Input dari keyboard
2. Input dari file
Cara input matrix: 2
Masukkan nama file:
../test/testcase4.txt
x1 = 14.444444444444446, x2 = 7.222222222222223, x3 = 10.0
```

c) SPL Matriks Balikan

```
1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Matriks balikan
4. Kaidah Cramer
Sub-menu yang ingin diakses: 3
1. Input dari keyboard
2. Input dari file
Cara input matrix: 2
Masukkan nama file:
../test/testcase4.txt
x1 = 14.444444444444443, x2 = 7.222222222222221, x3 = 10.000000000000002
```

d) SPL Kaidah Cramer

```
1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Matriks balikan
4. Kaidah Cramer
Sub-menu yang ingin diakses: 4
1. Input dari keyboard
2. Input dari file
Cara input matrix: 2
Masukkan nama file:
../test/testcase4.txt
x1 = 14.444444444444445, x2 = 7.222222222222222, x3 = 9.999999999999998
```

4.5 Studi dan Tes Kasus Interpolasi

Berikut adalah hasil eksperimen studi dan tes kasus interpolasi:

Tabel 4.5.1. Hasil Pengujian Studi dan Tes Kasus Interpolasi

No.	Soal dan Hasil Eksperimen
a.	1) Soal 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:

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

$x = 0.55$ $f(x) = ?$
$x = 0.85$ $f(x) = ?$
$x = 1.28$ $f(x) = ?$

2) Hasil Eksperimen
   a) Input *User*

```
MENU
1. Sistem Persamaan Linier
2. Determinan
3. Matriks Balikan
4. Interpolasi Polinom
5. Interpolasi Bicubic Spline
6. Regresi Linier dan Kuadratik Berganda
7. Scaling Gambar
8. Keluar
Menu yang ingin diakses: 4
1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Matriks balikan
4. Kaidah Cramer
Sub-menu yang ingin diakses: 2
1. Input dari keyboard
2. Input dari file
```

   b) x = 0.2

```
Masukkan nama file:
../test/testcase5ax1.txt
0.1 0.003
0.3 0.067
0.5 0.148
0.7 0.248
0.9 0.37
1.1 0.518
1.3 0.697
0.2
f(x) = -0.000000x^6 + 0.000000x^5 + 0.026042x^4 + 0.000000x^3 + 0.197396x^2 + 0.240000x - 0.022977
f(0.200000) = 0.032961
```

   c) x = 0.55

```
Masukkan nama file:
../test/testcase5ax2.txt
0.1 0.003
0.3 0.067
0.5 0.148
0.7 0.248
0.9 0.37
1.1 0.518
1.3 0.697
0.55
f(x) = -0.000000x^6 + 0.000000x^5 + 0.026042x^4 + 0.000000x^3 + 0.197396x^2 + 0.240000x - 0.022977
f(0.550000) = 0.171119
```

   d) x = 0.85

```
Masukkan nama file:
../test/testcase5ax3.txt
0.1 0.003
0.3 0.067
0.5 0.148
0.7 0.248
0.9 0.37
1.1 0.518
1.3 0.697
0.85
f(x) = -0.000000x^6 + 0.000000x^5 + 0.026042x^4 + 0.000000x^3 + 0.197396x^2 + 0.240000x - 0.022977
f(0.850000) = 0.337236
```

e) x = 1.28

```
Masukkan nama file:
../test/testcase5ax4.txt
0.1 0.003
0.3 0.067
0.5 0.148
0.7 0.248
0.9 0.37
1.1 0.518
1.3 0.697
1.28
f(x) = -0.000000x^6 + 0.000000x^5 + 0.026042x^4 + 0.000000x^3 + 0.197396x^2 + 0.240000x - 0.022977
f(1.280000) = 0.677542
```

b.

1) Soal

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:

Tanggal (desimal) = bulan + (tanggal / jumlah hari pada bulan tersebut)

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

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

b. 10/08/2022
c. 05/09/2022
d. Masukan user lainnya berupa tanggal (desimal) yang sudah diolah dengan asumsi prediksi selalu dilakukan untuk tahun 2022.

2) Hasil Eksperimen
   a) x = 7.516

```
Masukkan nama file:
../test/testcase5bx1.txt
6.567 12624.0
7.0 21807.0
7.258 38391.0
7.451 54517.0
7.548 51952.0
7.839 28228.0
8.161 35764.0
8.484 20813.0
8.709 12408.0
9.0 10534.0
7.516
f(x) = -140993.712249x^9 + 9372849.239101x^8 - 275474539.420669x^7 + 4695806315.428793x^6 - 51131876760.132810x^5 + 3685
50807175.535000x^4 - 1756810186361.356400x^3 + 5334203055240.578000x^2 - 9346993079173.438000x + 7187066071661.201000
f(7.516000) = 53566.808594
```

   b) x = 8.322

```
Masukkan nama file:
../test/testcase5bx2.txt
6.567 12624.0
7.0 21807.0
7.258 38391.0
7.451 54517.0
7.548 51952.0
7.839 28228.0
8.161 35764.0
8.484 20813.0
8.709 12408.0
9.0 10534.0
8.322
f(x) = -140993.712249x^9 + 9372849.239101x^8 - 275474539.420669x^7 + 4695806315.428793x^6 - 51131876760.132810x^5 + 3685
50807175.535000x^4 - 1756810186361.356400x^3 + 5334203055240.578000x^2 - 9346993079173.438000x + 7187066071661.201000
f(8.322000) = 36380.644531
```

   c) x = 9.166

```
Masukkan nama file:
../test/testcase5bx3.txt
6.567 12624.0
7.0 21807.0
7.258 38391.0
7.451 54517.0
7.548 51952.0
7.839 28228.0
8.161 35764.0
8.484 20813.0
8.709 12408.0
9.0 10534.0
9.166
f(x) = -140993.712249x^9 + 9372849.239101x^8 - 275474539.420669x^7 + 4695806315.428793x^6 - 51131876760.132810x^5 + 3685
50807175.535000x^4 - 1756810186361.356400x^3 + 5334203055240.578000x^2 - 9346993079173.438000x + 7187066071661.201000
f(9.166000) = -658969.265625
```

   d) x = 3.483

```
Masukkan nama file:
../test/testcase5bx4.txt
6.567 12624.0
7.0 21807.0
7.258 38391.0
7.451 54517.0
7.548 51952.0
7.839 28228.0
8.161 35764.0
8.484 20813.0
8.709 12408.0
9.0 10534.0
3.483
f(x) = -140993.712249x^9 + 9372849.239101x^8 - 275474539.420669x^7 + 4695806315.428793x^6 - 51131876760.132810x^5 + 3685
50807175.535000x^4 - 1756810186361.356400x^3 + 5334203055240.578000x^2 - 9346993079173.438000x + 7187066071661.201000
f(3.483000) = 3663469530.737793
```

c.

1) Soal

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

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.

2) Hasil Eksperimen

```
1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Matriks balikan
4. Kaidah Cramer
Sub-menu yang ingin diakses: 2
1. Input dari keyboard
2. Input dari file
Cara input matrix: 2
Masukkan nama file:
../test/testcase5c.txt
0.0 0.0
0.4 0.418884
0.8 0.507158
1.2 0.560925
1.6 0.583686
2.0 0.576651
0.5
f(x) = 0.236256x^5 - 1.421263x^4 + 3.237110x^3 - 3.552679x^2 + 2.035257x + 0.000000
f(0.500000) = 0.452651
```

4.6 Studi dan Tes Kasus Regresi Linear dan Kuadratik Berganda

Berikut adalah hasil eksperimen studi dan tes kasus regresi linear dan kuadratik berganda:

Tabel 4.6.1. Hasil Pengujian Studi dan Tes Kasus Regresi Linear dan Kuadratik Berganda

No.	Soal dan Hasil Eksperimen
a.	1) Soal

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. Silahkan terapkan model-model ini pada Multiple Quadratic Equation juga dan bandingkan hasilnya.

2) Hasil Eksperimen
   a) Input *User*

```
MENU
1. Sistem Persamaan Linier
2. Determinan
3. Matriks Balikan
4. Interpolasi Polinom
5. Interpolasi Bicubic Spline
6. Regresi Linier dan Kuadratik Berganda
7. Scaling Gambar
8. Keluar
Menu yang ingin diakses: 6
1. Regresi linier
2. Regresi kuadratik berganda
Sub-menu yang ingin diakses: 1
1. Input dari keyboard
2. Input dari file
Cara input matrix:
```

   b) Regresi Linear Berganda

```
1. Regresi linier
2. Regresi kuadratik berganda
Sub-menu yang ingin diakses: 1
1. Input dari keyboard
2. Input dari file
Cara input matrix: 2
Masukkan nama file untuk matriks:
../test/testcase6.txt
Choose regression method number:
1. Ordinary Least Squares (OLS)
2. Ridge Regression
1

f(x) = -3.507778139646689 + -0.0026249906767841225x1 + 7.989409758838717E-4x2 + 0.15415503029134925x3
f(xk) = 0.9384342282178126
Apakah output ingin disimpan ke file? (y/n):
```

   c) Regresi Kuadratik Berganda

```
1. Regresi linier
2. Regresi kuadratik berganda
Sub-menu yang ingin diakses: 2
1. Input dari keyboard
2. Input dari file
Cara input matrix: 2
Masukkan nama file untuk matriks:
../test/testcase6.txt
Choose regression method number:
1. Ordinary Least Squares (OLS)
2. Ridge Regression
1

f(x) = -2.6324959122055488E14 +
       1.666466695276919E11x1 + 2.500014559810019E12x2 + -2.868358718397004E12x3 +
       -3.760612424237101E8x1^2 + -1.0607211496413017E10x2^2 + 3.2135783724292584E11x3^2 +
       -4.1334254095630383E8x1x2 + -1.1727591798476677E9x1x3 + -2.762916180891589E10x2x3

f(xk) = -9.739003250722656E10
Apakah output ingin disimpan ke file? (y/n):
```

4.7 Studi dan Tes Kasus *Bicubic Spline*

Berikut adalah hasil eksperimen studi dan tes kasus *bicubic spline*:

Tabel 4.7.1. Hasil Pengujian Studi dan Tes Kasus *Bicubic Spline*

<table>
<tr><th>No.</th><th>Soal dan Hasil Eksperimen</th></tr>
<tr><td>a.</td><td>
1) Soal
$$\begin{pmatrix} 21 & 98 & 125 & 153 \\ 51 & 101 & 161 & 59 \\ 0 & 42 & 72 & 210 \\ 16 & 12 & 81 & 96 \end{pmatrix}$$
Tentukan nilai:
$f(0, 0)$ = ?
$f(0.5, 0.5)$ = ?
$f(0.25, 0.75)$ = ?
$f(0.1, 0.9)$ = ?

2) Hasil Eksperimen
a) Input *User*
MENU
1. Sistem Persamaan Linier
2. Determinan
3. Matriks Balikan
4. Interpolasi Polinom
5. Interpolasi Bicubic Spline
6. Regresi Linier dan Kuadratik Berganda
7. Scaling Gambar
8. Keluar
Menu yang ingin diakses: 5
interpolasi bicubic spline
1. Input dari keyboard
2. Input dari file
Cara input matrix:
b) x1 = 0, x2 = 0
Masukkan nama file untuk matriks:
../test/testcase7x1.txt
f(0.0, 0.0) = 101.0
Apakah output ingin disimpan ke file? (y/n):
c) x1 = 0.5, x2 = 0.5
Masukkan nama file untuk matriks:
../test/testcase7x2.txt
f(0.5, 0.5) = 88.3515625
Apakah output ingin disimpan ke file? (y/n):
d) x1 = 0.25, x2 = 0.75
</td></tr>
</table>

```
Masukkan nama file untuk matriks:
../test/testcase7x3.txt
f(0.25, 0.75) = 48.7391357421875
Apakah output ingin disimpan ke file? (y/n):
```

e) x1 = 0.1, x2 = 0.9

```
Masukkan nama file untuk matriks:
../test/testcase7x4.txt
f(0.1, 0.9) = 38.844678500000036
Apakah output ingin disimpan ke file? (y/n):
```

BAB V
KESIMPULAN, SARAN, DAN REFLEKSI

5.1 Kesimpulan

Matriks merupakan objek matematika yang sangat penting dan memiliki beragam aplikasi dalam berbagai bidang, baik di dunia akademik maupun dalam penerapannya di dunia nyata. Salah satu aplikasi utama matriks adalah dalam mencari solusi dari sistem persamaan linear (SPL). Dalam banyak kasus, berbagai masalah kompleks dapat dimodelkan ke dalam bentuk SPL, misalnya dalam perencanaan logistik, ekonomi, fisika, dan bidang lainnya. Metode penyelesaian SPL yang efisien sering kali melibatkan penggunaan operasi pada matriks seperti mencari invers matriks, reduksi baris, atau bahkan dengan memanfaatkan kaidah Cramer yang memungkinkan kita menyelesaikan SPL secara langsung melalui determinan.

Selain peran matriks dalam SPL, penggunaannya juga meluas ke bidang statistik dan *machine learning*. Misalnya, dalam statistik, matriks sering digunakan dalam perhitungan regresi linear maupun regresi polinomial yang memprediksi hubungan antara variabel-variabel melalui model matematis. Konsep regresi ini sangat bermanfaat dalam analisis data dan pembuatan model prediktif yang merupakan inti dari banyak aplikasi *machine learning*. Perhitungan-perhitungan ini juga dapat diterapkan pada data yang lebih kompleks melalui teknik interpolasi polinomial yang memperkirakan nilai antara titik-titik data yang diketahui.

Dalam bidang *computer graphics*, matriks memainkan peran penting dalam berbagai transformasi gambar. Salah satu aplikasinya adalah pada *bicubic spline interpolation*, sebuah metode yang digunakan untuk memperbesar gambar atau melakukan scaling tanpa kehilangan kualitas secara signifikan. Matriks digunakan untuk melakukan operasi seperti rotasi, translasi, dan *scaling* pada gambar digital yang merupakan dasar dalam teknologi *rendering* dan pengolahan citra modern.

Secara keseluruhan, aplikasi matriks tidak hanya terbatas pada pemodelan matematis atau penyelesaian masalah teoretis, tapi juga mencakup berbagai bidang praktis seperti statistik, *machine learning*, analisis data, dan grafika komputer. Melalui pemahaman yang mendalam tentang konsep-konsep matriks, kita dapat menyelesaikan berbagai permasalahan kompleks dengan lebih efisien dan akurat. Oleh karena itu, peran matriks dalam matematika dan aplikasinya sangatlah krusial dan relevan dalam kehidupan sehari-hari.

5.2 Saran

Penulis menyarankan untuk menyediakan *sample input* dan *sample output* pada setiap *test case* agar dapat dipastikan verifikasi *output* program telah memenuhi spesifikasi. Selain itu, akan sangat bermanfaat bila materi yang digunakan sebagai landasan pengerjaan tugas dilengkapi dengan referensi yang lebih jelas dan terverifikasi oleh asisten sehingga mahasiswa tidak mengalami kesulitan dan keraguan selama proses pengerjaan.

5.3 Refleksi

Sebagai refleksi pribadi penulis, pada awal proses pengerjaan, penulis berencana untuk membuat *graphical user interface* (GUI) pada akhir proses pengerjaan namun menjadi terabaikan akibat agenda-agenda lain, menyebabkan pengerjaan ini terabaikan. Penulis juga kurang melakukan testing sehingga ditemukan banyak bug di waktu-waktu terakhir. Menurut penulis, terdapat beberapa hal yang dapat ditambahkan, misal mencari basis suatu matriks, untuk mempermudah dalam pengerjaan komponen-komponen lainnya.

Di sisi lain, melalui pengerjaan tugas ini, penulis mendapatkan pemahaman yang lebih mendalam terhadap berbagai konsep matriks, operasi-operasinya, serta berbagai aplikasinya. Dengan proses ini, penulis sangat terbantu dalam memperkuat dasar-dasar konsep aljabar linear dan geometri serta mendapatkan wawasan lebih luas terhadap bagaimana teori-teori tersebut dapat diterapkan dalam pemecahan masalah nyata.

DAFTAR PUSTAKA

Strang, G. (2006) .*"Matrices and Gaussian Elimination," Linear Algebra and Its Applications, Fourth Edition*. United States of America: Cengage Learning, 2006. pp. 45.

Madematika (2017). Pengertian Minor, Kofaktor, Matriks Kofaktor, dan Adjoin Matriks. [Online] Tersedia: https://www.madematika.id/2017/08/pengertian-minor-kofaktor-matriks.html [Diakses 20 Oktober 2024].

Make42 (2019). *Multivariate Quadratic Regression*. [Online] Tersedia: https://math.stackexchange.com/questions/3155866/multivariate-quadratic-regression [Diakses 14 Oktober 2024].

Munir, R. (2010). Interpolasi Polinom (Bagian 1). [Online] Tersedia: https://informatika.stei.itb.ac.id/~rinaldi.munir/MetNum/2010-2011/Interpolasi%20Polinom.pdf [Diakses 20 Oktober 2024]

Munir, R. (2023). Sistem Persamaan Linear (Bagian 1: Metode Eliminasi Gauss). [Online] Tersedia: https://informatika.stei.itb.ac.id/~rinaldi.munir/AljabarGeometri/2023-2024/Algeo-03-Sistem-Persamaan-Linier-2023.pdf [Diakses 20 Oktober 2024].

Munir, R. (2023). Sistem Persamaan Linear (Bagian 2: Tiga Kemungkinan Solusi Sistem Persamaan Linear). [Online] Tersedia: https://informatika.stei.itb.ac.id/~rinaldi.munir/AljabarGeometri/2023-2024/Algeo-04-Tiga-Kemungkinan-Solusi-SPL-2023.pdf [Diakses 20 Oktober 2024].

Munir, R. (2024). Sistem Persamaan Linear (Bagian 3: Metode Eliminasi Gauss-Jordan). [Online] Tersedia: https://informatika.stei.itb.ac.id/~rinaldi.munir/AljabarGeometri/2023-2024/Algeo-05-Sistem-Persamaan-Linier-2-2023.pdf [Diakses 20 Oktober 2024].

Rahma, A.N., Safitri, E., Rahmawati (2019). Determinan Matriks *FLScirc*$_r$ Bentuk Khusus n x n, n ≥ 3 Menggunakan Metode Kondensasi Chio. *Jurnal Sains Matematika dan Statistika. 5(1):* 24.

Sinha, Priyanka (2013). Multivariate Polynomial Regression in Data Mining: Methodology, Problems and Solutions. *International Journal of Scientific & Engineering Research. 4(12):* 964.

Profematika (2019). Eliminasi Gauss dan Contoh Penerapannya. [Online] Tersedia: https://www.profematika.com/eliminasi-gauss-dan-contoh-penerapannya/ [Diakses 20 Oktober 2024]

Syaj, B.H., Armin, A.P., Yunanda, A.B. (2019). Rekonstruksi Citra pada Super Resolusi Menggunakan Interpolasi Bicubic. *INTEGER Journal of Information Technology. 4(2)*.

Taylor, Sebastian (2024). *Multiple Linear Regression.* [Online] Tersedia: https://corporatefinanceinstitute.com/resources/data-science/multiple-linear-regression/#:~:text=Multiple%20linear%20regression%20refers%20to%20a%20statistical%20technique%20that%20uses,variable%20in%20the%20total%20variance [Diakses 14 Oktober 2024].

Wikipedia (2017). Kaidah Creamer. [Online] Tersedia: https://id.wikipedia.org/wiki/Kaidah_Cramer [Diakses 11 Oktober 2024].

LAMPIRAN

a. Repository : https://github.com/BP04/Algeo01-23013

b. Video : https://youtu.be/ljSp8kCUlr4