LAPORAN TUGAS BESAR 1 IF2123
ALJABAR LINIER DAN GEOMETRI

Dosen Pengampu :
Ir. Rila Mandala, M.Eng., Ph.D.

Asisten Pembimbing :
Ikhwan Al Hakim

Disusun oleh :
Kelompok 24 (Sonny Angel)

Muhammad Fithra Rizki	13523049
Rafif Farras	13523095
Ahmad Wicaksono	13523121

PROGRAM STUDI TEKNIK INFORMATIKA
SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA
INSTITUT TEKNOLOGI BANDUNG
JL. GANESA 10, BANDUNG 40132

2024

KATA PENGANTAR

Puji syukur penulis panjatkan kepada Tuhan Yang Maha Esa atas segala rahmat dan hidayah-Nya sehingga penulis dapat menyusun laporan tugas besar 1 ini. Laporan ini menjadi bagian dari tugas mata kuliah Aljabar Linier dan Geometri yang menjadi dasar penting untuk memahami konsep-konsep yang bersifat fundamental dalam ranah matematika dan pemrograman.

Melalui penyusunan laporan ini, penulis berupaya penuh dalam menyajikan materi secara komprehensif yang mencakup topik-topik penting, di antara lain determinan, matriks balikan, hingga regresi linier dan kuadratik berganda. Hal ini membuat penulis menyadari bahwa pentingnya pemahaman yang kuat terhadap topik yang dibahas pada laporan ini, terutama dalam penerapan pada bidang-bidang yang penulis tekuni, seperti komputasi, analisis data, serta ilmu-ilmu teknik.

penulis juga mengucapkan rasa terima kasih yang sebesar-besarnya kepada dosen pengampu Aljabar Linier dan Geometri, Pak Ir. Rila Mandala, M.Eng., Ph.D., serta kepada asisten pembimbing penulis, Kak Ikhwan Al Hakim, yang telah memberikan bimbingan dan dukungan selama proses pengerjaan tugas besar dan penyusunan laporan ini. penulis sangat berharap laporan ini dapat menjadi suatu kontribusi dari penulis yang bermanfaat, baik untuk diri penulis sendiri maupun untuk para pembaca yang ingin lebih mendalami ilmu ini.

Akhir kata, semoga laporan ini dapat menjadi acuan yang berguna dalam proses pembelajaran dan bisa menjadi pendorong untuk penulis maupun pembaca untuk bisa mengeksplorasi lebih jauh seputar materi Aljabar Linier dan Geometri kedepannya.

Bandung, 24 Oktober 2024

Muhammad Fithra Rizki (13523049)

Rafif Farras (13523095)

Ahmad Wicaksono (13523121)

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB I
DESKRIPSI MASALAH

1.1 Sistem Persamaan Linier (SPL)

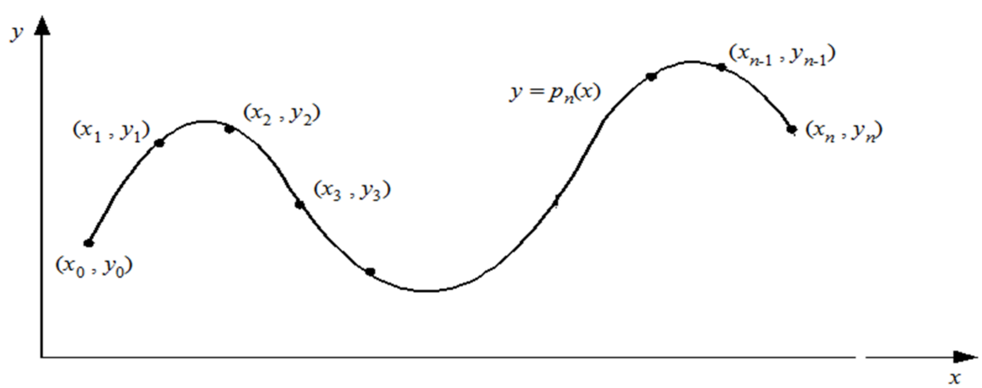
Sistem persamaan linier (SPL) banyak ditemukan di dalam bidang sains dan rekayasa. Andstrea sudah mempelajari berbagai metode untuk menyelesaikan SPL, termasuk menghitung determinan matriks. Sembarang SPL dapat diselesaikan dengan beberapa metode, yaitu metode eliminasi Gauss, metode eliminasi Gauss-Jordan, 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 Eliminasi Gauss dilakukan dengan matriks eselon baris dan eliminasi Gauss-Jordan dengan matriks eselon baris tereduksi

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 $i = 0, 1, 2, \ldots, n$.



Gambar 1.2 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 $i = 0, 1, 2, \ldots, n$, akan diperoleh n buah sistem persamaan lanjar dalam $a_0, a_1, a_2, \ldots, a_n$,

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

Solusi sistem persamaan lanjar ini, yaitu nilai a_0, a_1, $\ldots$, 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

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

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

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

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 X^2

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 :

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

N menandakan jumlah peubah, terdapat 2 variabel linier yaitu u_i dan v_i, 2 variabel kuadrat yaitu u_i^2 dan v_i^2, dan 1 variabel interaksi yaitu uv. Untuk setiap n-peubah, akan terdapat 1 konstan N (Terlihat di bagian atas kiri gambar), n variabel linier, n variabel kuadrat, dan C_2^n variabel linier (dengan syarat n > 1). Tentu dengan bertambahnya peubah n, ukuran matriks akan bertumbuh lebih besar dibandingkan regresi linier berganda tetapi solusi tetap bisa didapat dengan menggunakan SPL.

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.

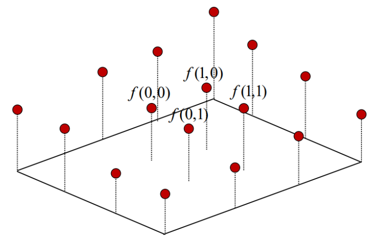
Dalam pemrosesan menggunakan interpolasi bicubic spline digunakan 16 buah titik, 4 titik referensi utama di bagian pusat, dan 12 titik di sekitarnya sebagai aproksimasi turunan dari keempat titik referensi untuk membagun permukaan bikubik. Bentuk pemodelannya adalah sebagai berikut.

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

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

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

Solve: a_{ij}



Gambar 1.4.1 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.

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

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

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

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

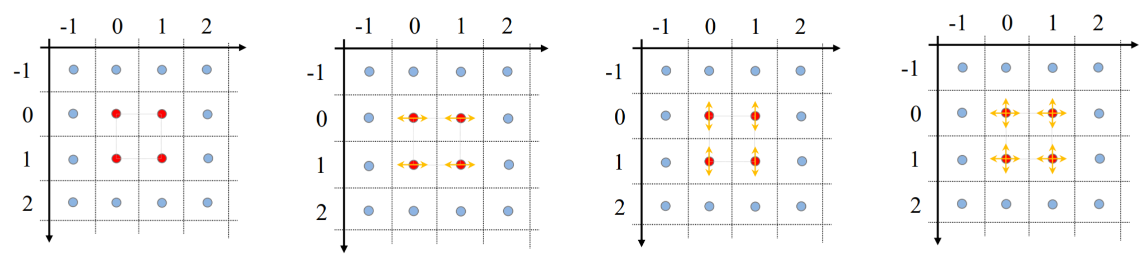
Dengan menggunakan nilai fungsi dan turunan berarah tersebut, dapat terbentuk sebuah matriks solusi X yang membentuk persamaan penyelesaian sebagai berikut.

$$y = Xa$$

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

Perlu diketahui bahwa elemen pada matriks X adalah nilai dari setiap komponen koefisien a_{ij} yang diperoleh dari persamaan fungsi maupun persamaan turunan yang telah dijelaskan sebelumnya. Sebagai contoh, elemen matriks X pada baris 8 kolom ke 2 adalah koefisien dari a_{10} pada ekspansi sigma untuk $f_x(1, 1)$ sehingga diperoleh nilai konstanta $1 \times 1^{1-1} \times 1^0 = 1$, sesuai dengan isi matriks X.

Nilai dari vektor a dapat dicari dari persamaan y = Xa, lalu vektor a tersebut digunakan sebagai nilai variabel dalam f(x, y), sehingga terbentuk fungsi interpolasi bicubic sesuai model. 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 1.4.2 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 Metode Eliminasi Gauss

Metode eliminasi gauss adalah suatu metode yang digunakan dalam operasi nilai-nilai matriks dan bertujuan untuk menyederhanakan suatu matriks. Metode eliminasi gauss memanfaatkan metode eliminasi dengan cara menghilangkan atau mengurangi jumlah variabel yang ada untuk mendapat nilai variabel bebas.

Metode eliminasi gauss mengubah persamaan linear menjadi suatu bentuk matriks teraugmentasi yang kemudian akan dibentuk menjadi matriks eselon baris melalui suatu operasi yang disebut Operasi Baris Elementer (OBE). Setelah dibuat menjadi bentuk matriks eselon baris, matriks kemudian diselesaikan dengan metode substitusi balik untuk mendapat nilai variabel dari Sistem Persamaan Linear.

$$\begin{bmatrix} 0 & 1 & * & * & * & * & * & * & * & * \\ 0 & 0 & 0 & 1 & * & * & * & * & * & * \\ 0 & 0 & 0 & 0 & 1 & * & * & * & * & * \\ 0 & 0 & 0 & 0 & 0 & 1 & * & * & * & * \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & * \end{bmatrix}$$

Gambar 2.1.1 Matriks Eselon Baris

Solusi dari Sistem Persamaan Linear yang didapat memiliki kemungkinan tiga bentuk, yakni solusi unik/tunggal, solusi banyak/tak hingga, dan tidak memiliki solusi.

Solusi unik/tunggal didapat jika satu utama yang terdapat pada matriks berada di tepat diagonal matriks.

$$\left[\begin{array}{ccc|c} 1 & 1 & 1 & 0 \\ 2 & 3 & 1 & 1 \\ 3 & 1 & 2 & 1 \end{array}\right] \xrightarrow{\text{Eliminasi Gauss}} \left[\begin{array}{ccc|c} 1 & 1 & 1 & 0 \\ 0 & 1 & -1 & 1 \\ 0 & 0 & 1 & 1 \end{array}\right]$$

Gambar 2.1.2 Solusi Unik/Tunggal dari Metode Eliminasi Gauss

Solusi banyak/tak hingga didapat jika terdapat baris akhir pada matriks yang memiliki nilai 0 atau jika ditulis dalam persamaan linier akan membentuk $0x_1 + 0x_2 + \ldots + 0x_n = 0$. Selain itu, jumlah kolom matriks augmented berbeda terdapat lebih 1 dari jumlah baris pada matriks augmented, solusi banyak/tak hingga dapat terjadi.

$$\left[\begin{array}{ccc|c} 1 & 1 & 2 & 4 \\ 2 & -1 & 1 & 2 \\ 1 & 2 & 3 & 6 \end{array}\right] \xrightarrow{\text{Eliminasi Gauss}} \left[\begin{array}{ccc|c} 1 & 1 & 2 & 4 \\ 0 & 1 & 1 & 2 \\ 0 & 0 & 0 & 0 \end{array}\right]$$

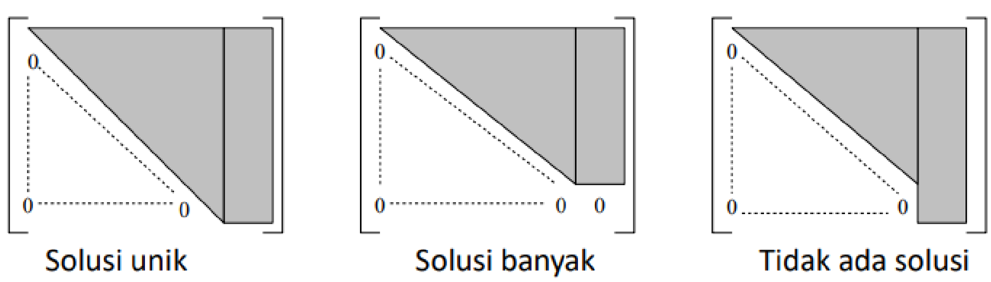
Gambar 2.1.3 Solusi Banyak/Tak Hingga dari Metode Eliminasi Gauss

Sistem persamaan linier bisa tidak memiliki solusi jika matriks augmented memiliki baris akhir yang memiliki nilai 0 pada setiap x nya, tetapi memiliki hasil bukan 0. Dalam bentuk persamaan linier akan membentuk $0x_1 + 0x_2 + \ldots + 0x_n = a$ dengan nilai $a \neq 0$.

$$\left[\begin{array}{ccc|c} 1 & 1 & 2 & 4 \\ 2 & -1 & 1 & 2 \\ 1 & 2 & 3 & 7 \end{array}\right] \xrightarrow{\text{Eliminasi Gauss}} \left[\begin{array}{ccc|c} 1 & 1 & 2 & 4 \\ 0 & 1 & 1 & 2 \\ 0 & 0 & 0 & 1 \end{array}\right]$$

Gambar 2.1.4 Persamaan Tidak Memiliki Solusi dari Metode Eliminasi Gauss

Tiga kemungkinan solusi ini membentuk suatu pola bangun datar yang sama untuk masing-masing solusi. Sehingga terdapat suatu bentuk akhir pada solusi dari persamaan Ax = b dengan gambaran sebagai berikut.



Gambar 2.1.5 Gambar Akhir Solusi dari Sistem Persamaan Linier

2.2 Metode Eliminasi Gauss-Jordan

Metode eliminasi gauss-jordan merupakan metode lanjutan dari metode eliminasi gauss. Pada metode ini, Operasi Baris Elementer diterapkan untuk menghasilkan suatu matriks eselon tereduksi. Namun, terdapat tahap lanjutan setelah terbentuk matriks tersebut. Metode ini terbagi menjadi dua fase, yakni fase maju dan fase mundur. Fase maju memiliki cara kerja yang sama dengan metode eliminasi gauss dan menghasilkan nilai 0 dibawah satu utama. Hal yang membedakan adalah fase mundur yang dimana fase mundur akan membentuk nilai 0 di atas satu utama dan membuat matriks augmented tersebut langsung memiliki nilai variabel tanpa melakukan substitusi balik (jika persamaan linier memiliki solusi unik).

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

Gambar 2.2 Metode Eliminasi Gauss-Jordan

Sama seperti metode eliminasi gauss, metode eliminasi gauss-jordan juga dapat menghasilkan tiga kemungkinan solusi, yakni solusi unik/tunggal, solusi banyak/tak hingga, dan tidak memiliki solusi.

2.3 Determinan

Determinan adalah suatu bilangan yang didapat dari suatu matriks persegi (n x n) yang memiliki berbagai aplikasi penting dalam aljabar linier, termasuk pada penyelesaian Sistem Persamaan Linier (SPL), invers matriks, serta karakteristik transformasi linier.

Determinan memegang peranan dalam mendapatkan informasi seputar sifat-sifat matriks, seperti apakah suatu matriks memiliki invers atau tidak dan lainnya. Untuk matriks dengan ukuran 2x2, determinan nya dapat ditentukan dengan cara berikut.

$$\det \begin{bmatrix} a & b \\ c & d \end{bmatrix} \equiv \begin{vmatrix} a & b \\ c & d \end{vmatrix} \equiv a\,d - b\,c.$$

Gambar 2.3 Determinan Matriks 2x2

Untuk matriks yang memiliki ukuran lebih besar 2x2, determinan dapat dibentuk dengan menggunakan metode ekspansi kofaktor dan metode eliminasi gauss.

2.4 Matriks Balikan

Matriks balikan atau invers matriks adalah suatu bentuk matriks yang jika dikalikan dengan asal matriks itu sendiri akan menghasilkan matriks identitas. Matriks balikan dapat dibentuk jika matriks tersebut memiliki nilai determinan tidak sama dengan 0. Balikan suatu matriks (contoh : matriks A) dapat ditulis dalam bentuk A^{-1} . Untuk matriks 2x2, bentuk balikan matriks dapat dicari dengan suatu cara berikut.

$$A = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \longrightarrow A^{-1} = \frac{1}{ad - bc} \begin{bmatrix} d & -b \\ -c & a \end{bmatrix}$$

Gambar 2.4 Matriks Balikan 2x2

Jika sebuah matriks dapat dicari inversnya, matriks tersebut dapat disebut invertible atau dapat diinversi. Jika matriks tersebut tidak dapat dicari inversnya atau memiliki nilai determinan sama dengan 0, maka matriks tersebut disebut matriks singular.

2.5 Matriks Kofaktor

Matriks kofaktor merupakan matriks yang dimana setiap elemennya merupakan kofaktor dari elemen-elemen matriks asal. Kofaktor sendiri adalah bilangan yang dihasilkan dari minor matriks dan kemudian dikalikan dengan faktor tanda. Matriks kofaktor digunakan dalam berbagai perhitungan, termasuk mencari determinan dan invers matriks.

Kofaktor menuruti suatu aturan, yaitu $(-1)^{(i+j)}$ dengan i adalah baris dan j adalah kolom. Kofaktor juga biasanya ditandakan dengan C_{ij}.

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

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

Gambar 2.5 Matriks Kofaktor

2.6 Matriks Adjoin

Matriks adjoin merupakan matriks yang diperoleh dari transpose matriks kofaktor. Matriks ini digunakan dalam berbagai aplikasi, terutama dalam perhitungan invers matriks. Matriks adjoin sangat penting karena berperan dalam rumus invers matriks.

Jika A merupakan suatu matriks persegi berukuran nxn, maka adjoin (adj(A)) dapat dicari dengan persamaan berikut.

adj (A) = transpose (kofaktor (A))

2.7 Kaidah *Cramer*

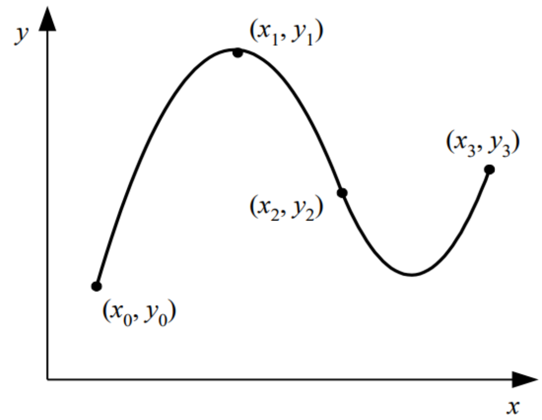
Kaidah Cramer juga menjadi salah satu alternatif metode dalam menyelesaikan sistem persamaan linear. Penyelesaian SPL menggunakan kaidah Cramer adalah dengan mencari nilai determinan dari matriks koefisien dan determinan matriks yang dihasilkan dengan menggantikan setiap kolom matriks koefisien A dengan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

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

Gambar 2.7 Solusi melalui Kaidah Cramer

2.8 Interpolasi Polinom

Interpolasi polinom adalah metode dalam matematika untuk menemukan sebuah polinom yang melewati sekumpulan titik data tertentu. Dengan menggunakan interpolasi polinom, kita dapat membentuk fungsi polinom yang mendekati atau tepat melalui sejumlah titik data yang diketahui, dan memprediksi nilai di antara titik-titik tersebut.

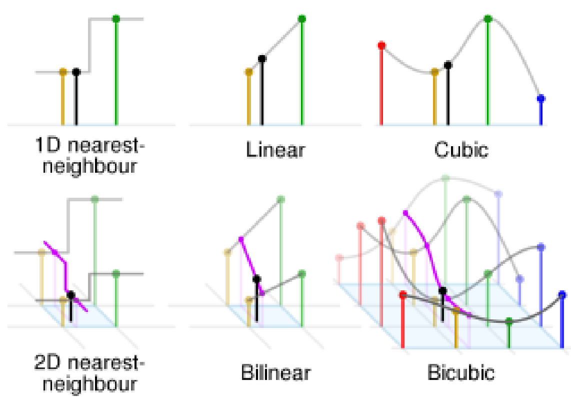


Gambar 2.8 Contoh Interpolasi Polinom

2.9 Interpolasi *Bicubic Spline*

Interpolasi *bicubic spline* merupakan suatu metode interpolasi dua dimensi yang berfungsi untuk memperkirakan nilai di antara titik-titik data pada grid atau permukaan. Dengan metode ini, konsep interpolasi *cubic spline* diperluas ke dalam dua dimensi sehingga biasa digunakan untuk memperhalus gambar atau data permukaan yang memerlukan interpolasi dengan hasil halus.

Interpolasi *bicubic spline* membentuk fungsi interpolasi f (x,y) dari gabungan polinomial kubik dalam dua variabel x dan y, yang memastikan bahwa interpolasi berjalan halus tidak hanya di titik-titik data tetapi juga di antara titik tersebut.



Gambar 2.9 Bentuk-Bentuk Interpolasi

2.10 Regresi Linier dan Kuadratik Berganda

Regresi linier berganda merupakan suatu metode statistika yang berfungsi untuk pemodelan hubungan antara satu variabel dependen dengan dua atau lebih variabel independen. Metode ini adalah pengembangan dari regresi linier sederhana yang hanya menggunakan satu variabel independen. Dalam regresi linier berganda, dapat diasumsikan bahwa variabel dependen memiliki hubungan yang linier dengan variabel independen lainnya.

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_n X_n + \epsilon$$

Regresi kuadratik berganda adalah suatu model dengan memperhitungkan pengaruh dari kuadratik variabel-variabel independen terhadap variabel dependen. Model ini memungkinkan perluasan dari regresi linier berganda dengan penambahan kuadrat dari variabel-variabel independen atau interaksi antara dua variabel independen.

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_1^2 + \beta_4 X_2^2 + \beta_5 X_1 X_2 + \cdots + \epsilon$$

BAB III

IMPLEMENTASI PUSAKA DAN PROGRAM

3.1 Implementasi Pusaka

Pada tugas besar ini, perlu dibuat pustaka berbentuk *Abstract Data Type* (ADT) Matriks yang memuat fungsi-fungsi primitif untuk menyelesaikan permasalahan yang dibahas. Fungsi-fungsi ini juga menyelesaikan permasalahan dengan beberapa metode seperti metode Gauss, Gauss-Jordan, Invers, dan lain-lain. Masalah yang akan penulis bahas seputar Sistem Persamaan Linier (SPL), Interpolasi Polinomial, Regresi Berganda, serta *Bicubic Spline Interpolation*.

3.1.1 MatriksInput.java

MatriksInput.java berisi fungsi-fungsi yang dapat menerima masukan matriks, dapat berupa input manual ataupun file (.txt).

1. Atribut

Nama Atribut	**Fungsi**
input	Membuat Scanner untuk melakukan input

2. Konstruktor

   -

3. Metode

Nama Metode	**Fungsi**
normalMatrix()	Input matriks biasa dari pengguna
augmentedMatrix()	Input matriks augmented (matriks utama dan hasil) dari pengguna
fileMatrix()	Membaca file eksternal berisi matriks dan mengembalikan dalam bentuk MatrixOperasi
readInterpolasiKeyboard()	Input titik x dan y dari pengguna untuk Interpolasi Polinom

regresiMatrix(boolean isQuadratic)	Input peubah x dan y untuk regresi linier atau kuadratik
hilbertMatrix()	Input matriks hilbert berdasarkan jumlah kolom dan input hasil SPL nya

3.1.2 MatrixOperasi.java

MatriksOperasi.java berisi operasi-operasi yang akan digunakan untuk menjalankan fungsi-fungsi yang ada.

1. Atribut

Nama Atribut	**Fungsi**
matrix	Menyimpan matriks dengan bentuk double[][]
row	Menyimpan baris dalam integer
col	Menyimpan kolom dalam integer
MARK	Menyimpan nilai default pada khususnya pada matriks

2. Konstruktor

Nama Konstruktor	**Fungsi**
MatrixOperasi(double contents[][], int rows, int cols)	Inisiasi matriks dengan bentuk array 2D dengan nilai yang sudah didefinisi
MatrixOperasi(int rows, int cols)	Inisiasi matriks dengan bentuk array 2D dengan nilai yang belum didefinisi

3. Metode

Nama Metode	**Fungsi**
setElmt(int i, int j, double elmt)	Menetapkan elemen elmt pada posisi [i][j]
getColEff()	Mencari nilai kolom efektif
getRowEff()	Mencari nilai baris efektif
getLastIdxCol()	Mencari index terakhir kolom

getElmt(int i, int j)	Mencari nilai elemen pada posisi [i][j]
sumCol(MatrixOperasi m, int i)	Mencari nilai sum pada satu kolom
sumMultiplyCol(MatrixOperasi m, int i, int j)	Mencari nilai sum dari perkalian dua kolom
swapRow (MatrixOperasi m, int row1, int row2)	Menukar dua baris matriks
addRow (MatrixOperasi m, double[] newRow)	Menambah baris baru pada matriks
isSquare (MatrixOperasi m)	Mengecek apakah matriks berbentuk persegi
createIdentity(int n)	Membuat matriks identitas nxn
isIdentity(MatrixOperasi m)	Mengecek apakah matriks identitas
FirstElementNot0Coll(MatrixOperasi m, int startrowIdx, int colIdx)	Mengecek apakah elemen pertama matriks bernilai nol
solutionType(MatrixOperasi matrix)	Menentukan tipe solusi matriks augmented (tidak ada, tak hingga, atau unik)
KolomNol(MatrixOperasi m, int idxCol)	Mengecek nilai 0 pada suatu kolom matriks
parametrik(MatrixOperasi m)	Mencari solusi parametrik matriks
backSubstitution(MatrixOperasi matrix, double[] X)	Membuat segitiga atas pada matriks yang telah direduksi
gaussElimination(MatrixOperasi matrix)	Melakukan eliminasi gauss pada matriks
gaussJordanElimination(MatrixOperasi A)	Melakukan eliminasi gauss-jordan pada matriks
detKofaktorRC(MatrixOperasi m, int row, int col)	Mencari determinan dari matriks minor
matriksKofaktor (MatrixOperasi m)	Membentuk matriks kofaktor
Adjoin (MatrixOperasi m)	Membentuk matriks adjoin dari matriks kofaktor

multiplyMatrix(MatrixOperasi m1, MatrixOperasi m2)	Melakukan perkalian dua matriks

3.1.3 MatriksOutput.java

MatriksOutput.java berisi fungsi-fungsi untuk membuat output/print matriks yang telah dioperasikan atau diinput.

1. Atribut

Nama Atribut	Fungsi
result	Membuat String[] dari matriks

2. Konstruktor

   -

3. Metode

Nama Metode	Fungsi
printMatrix(MatrixOperasi m)	Melakukan print pada nilai matriks
WriteStringArrayToFile(String[] content, String filename)	Membuat sebuah string dalam bentuk array ke sebuah file
saveFile(String[] results)	Menyimpan output dalam bentuk file
matrixToString(MatrixOperasi matrix)	Menyimpan output matriks dalam bentuk string
matrixToStringArray(MatrixOperasi m)	Menyimpan output matriks dalam bentuk string array

3.1.4 Main.java

Main.java berisi fungsi pusat untuk menjalankan seluruh fungsi lainnya, dengan kata lain menjadi fungsi yang menjalankan keseluruhan program.

1. Atribut

Nama Atribut	Fungsi
os	Untuk menyesuaikan fungsi dengan OS yang digunakan pengguna

input	Membuat Scanner untuk melakukan input

2. Konstruktor

   -

3. Metode

Nama Metode	**Fungsi**
clearConsole()	Melakukan pembersihan terminal setelah input dilakukan
main(String[] args)	Fungsi utama untuk menjalankan dan menggabungkan fungsi-fungsi lainnya yang ada pada program ini

3.1.5 SPL.java

SPL.java memiliki beberapa fungsi, yakni eliminasi gauss, eliminasi gauss-jordan, kaidah cramer, dan invers matriks yang berguna untuk mencari solusi dari suatu matriks yang berasal dari Sistem Persamaan Linier dan dapat membantu untuk menyelesaikan persoalan yang dibahas.

1. Atribut

   -

2. Konstruktor

   -

3. Metode

Nama Metode	**Fungsi**
gaussSPL(MatrixOperasi matrix)	Menyelesaikan SPL dengan metode eliminasi gauss dan menampilkan solusi
gaussJordanSPL(MatrixOperasi matrix)	Menyelesaikan SPL dengan metode eliminasi gauss jordan dan menampilkan solusi
cramerSPL(MatrixOperasi matrix)	Menyelesaikan SPL dengan kaidah cramer dan menampilkan solusi
inversSPL(MatrixOperasi matrix)	Menyelesaikan SPL dengan mencari invers matriks dan menampilkan solusi

3.1.6 Determinan.java

Determinan.java dapat mengembalikan nilai determinan dengan metode kofaktor dan reduksi baris.

1. Atribut

   -

2. Konstruktor

   -

3. Metode

Nama Metode	**Fungsi**
determinanKofaktor(MatrixOperasi m)	Mencari determinan dengan metode ekspansi kofaktor
determinanReduksi(MatrixOperasi m)	Mencari determinan dengan metode reduksi baris
handleDeterminantCases(MatrixOperasi m)	Menyimpan determinan dalam bentuk string yang akan disimpan di file

3.1.7 Invers.java

Invers.java dapat mengembalikan nilai invers matriks dengan metode adjoin dan dengan memanfaatkan matriks identitas.

1. Atribut

   -

2. Konstruktor

   -

3. Metode

Nama Metode	**Fungsi**
inversAdjoin(MatrixOperasi m)	Mencari nilai invers matriks dengan metode adjoin
inversIdentitas(MatrixOperasi m)	Mencari nilai invers matriks dengan memanfaatkan matriks identitas

3.1.8 InterpolasiPolinomial.java

InterpolasiPolinomial.java akan membentuk polinomial interpolasi dari titik-titik yang ada dan membuat nilai taksiran untuk suatu x.

1. Atribut

   -

2. Konstruktor

   -

3. Metode

Nama Metode	**Fungsi**
InterPolim(MatrixOperasi matrix, Boolean m)	Membentuk solusi koefisien polinomial dengan membentuk matriks augmented dari nilai x dan y
runInterpolasi(MatrixOperasi m, double x)	Menghitung dan mencetak hasil interpolasi polinomial berdasarkan input x dan koefisien yang dihitung
interPolimVal(double[] values, double t)	Menghitung nilai interpolasi polinomial dari satu variabel

3.1.9 Regresi.java

Regresi.java memiliki fungsi yang melakukan operasi regresi linier berganda dan regresi kuadratik berganda.

1. Atribut

   -

2. Konstruktor

   -

3. Metode

Nama Metode	**Fungsi**
regresiLinearKeyboard(MatrixOperasi m)	Menghitung regresi linier berdasarkan input pengguna melalui keyboard dan menampilkan persamaan regresi serta hasil estimasi

regresiLinearFile(MatrixOperasi m)	Menghitung regresi linier berdasarkan input pengguna melalui input file dan menampilkan persamaan regresi serta hasil estimasi
regresiLinearQuadraticKeyboard(MatrixOperasi m)	Menghitung regresi kuadratik berdasarkan input dari keyboard
regresiLinearQuadraticFile(MatrixOperasi m)	Menghitung regresi kuadratik berdasarkan input dari file

3.1.10 BicubicInterpolation.java

BicubicInterpolation.java memiliki fungsi yang membentuk solusi dari fungsi y = Xa untuk nilai f (x,y).

1. Atribut

   -

2. Konstruktor

   -

3. Metode

Nama Metode	Fungsi
createMatrixF()	Membuat matriks 4x16 untuk bicubic dengan fungsi f (x,,y)
createMatrixFx()	Membuat matriks 4x16 untuk bicubic dengan fungsi f_x (x,y)
createMatrixFy()	Membuat matriks 4x16 untuk bicubic dengan fungsi f_y (x,y)
createMatrixFxy()	Membuat matriks 4x16 untuk bicubic dengan fungsi f_{xy} (x,y)
createMatrixX()	Membuat matriks 16x16 yang merupakan gabungan dari fungsi-fungsi sebelumnya

bicubicInterpolation(MatrixOperasi inputMatrix)	Metode untuk melakukan interpolasi bicubic dan menampilkan hasil dari input x dan y untuk f (x,y)
bicubicVal(double[][] values, double x, double y)	Mencari array nilai interpolasi polinomial dari input values yang ada dengan y, array tersebut kemudian di interpolasi dengan x

3.1.11 ImageResizing.java

ImageResizing.java berfungsi untuk mengubah ukuran gambar axb dan di dilatasi sebesar 1,5 kali untuk a dan 2x untuk b dengan format jpg atau png.

1. Atribut

   -

2. Konstruktor

   -

3. Metode

Nama Metode	**Fungsi**
resizeImage(BufferedImage originalImage, double scaleX, double scaleY)	Mengubah size gambar sesuai dengan skala yang diminta
blok4x4(BufferedImage image, double x, double y)	Mengambil blok 4x4 disekitar titik (x,y) dan membentuk blok warna RGB
bicubicInterpolationRGB(double[][][] pixels, double x, double y)	Interpolasi bicubic untuk masing masing warna merah, hijau, dan biru
resizing(String file)	Membuat input dari file pada test/input, mengoperasikan resizeImage, dan menyimpan pada test/output

3.1.12 Pemanis.java

1. Atribut

   -

2. Konstruktor

   -

3. Metode

Nama Metode	Fungsi
public static void sonnyangel()	Memberikan teks bertuliskan “sonny angel” untuk mempercantik terminal program awal
public static void spl()	Memberikan teks bertuliskan “spl solver” untuk mempercantik terminal program pada menu SPL
public static void deter()	Memberikan teks bertuliskan “determinan” untuk mempercantik terminal program pada menu determinan
public static void inve()	Memberikan teks bertuliskan “invers” untuk mempercantik terminal program pada menu matriks balikan
public static void inter()	Memberikan teks bertuliskan “interpolasi polinom” untuk mempercantik terminal program pada menu interpolasi polinom
public static void bicub()	Memberikan teks bertuliskan “bicubic” untuk mempercantik terminal program pada menu interpolasi bicubic spline
public static void regre()	Memberikan teks bertuliskan “regresi” untuk mempercantik terminal program pada menu regresi berganda
public static void gambar()	Memberikan teks bertuliskan “interpolasi gambar” untuk mempercantik terminal program menu interpolasi gambar
public static void tengkyu()	Memberikan teks bertuliskan “thank you” untuk mempercantik terminal program keluar
public static void nofitur()	Memberikan teks bertuliskan “feature is unavailable” untuk mempercantik terminal program ketika menginput nomor diluar menu

BAB IV
EKSPERIMEN

Tabel 4 Hasil Eksekusi Program

No.	Soal dan Hasil Eksekusi
1a	$A = \begin{bmatrix} 1 & 1 & -1 & -1 \\ 2 & 5 & -7 & -5 \\ 2 & -1 & 1 & 3 \\ 5 & 2 & -4 & 2 \end{bmatrix}, \quad b = \begin{bmatrix} 1 \\ -2 \\ 4 \\ 6 \end{bmatrix}$
	1. Hasil Menggunakan Gauss Elimination Hasil Matrix Setelah Gauss Elimination 1,0000 1,0000 -1,0000 -1,0000 1,0000 0,0000 1,0000 -1,6667 -1,0000 -1,3333 0,0000 0,0000 1,0000 -1,0000 1,0000 0,0000 0,0000 0,0000 0,0000 1,0000 Solusi tidak ada. 2. Hasil Menggunakan Gauss-Jordan Elimination Hasil Matrix Setelah Gauss Jordan Elimination 1,0000 0,0000 0,0000 0,6667 1,6667 0,0000 1,0000 0,0000 -2,6667 0,3333 0,0000 0,0000 1,0000 -1,0000 1,0000 0,0000 0,0000 0,0000 0,0000 1,0000 Solusi tidak ada. 3. Hasil Menggunakan Matriks Balikan SPL tidak dapat diselesaikan dengan matriks balikan karena nilai balikan matriks tidak terdefinisi. 4. Hasil Menggunakan Kaidah Cramer SPL tidak dapat diselesaikan dengan metode cramer.

1b

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

1. Hasil Menggunakan Gauss Elimination

```
Hasil Matrix Setelah Gauss Elimination
1,0000 -1,0000 0,0000 0,0000 1,0000 3,0000
0,0000 1,0000 0,0000 -1,5000 -0,5000 1,5000
0,0000 0,0000 0,0000 1,0000 -1,0000 -1,0000
0,0000 0,0000 0,0000 1,0000 -1,0000 -1,0000

Solusi banyak (parametrik):
x1 = 3,000 + 1,000x2 - 1,000x5
x2 = 1,500 + 1,500x4 + 0,500x5
x3 = c
x4 = -1,000 + 1,000x5
x5 = e
```

Jika disederhanakan, maka:
x1 = 3,000 + 1,000e
x2 = 2,000e
x3 = c
x4 = -1,000 + e
x5 = e

2. Hasil Menggunakan Gauss-Jordan Elimination

```
Hasil Matrix Setelah Gauss Jordan Elimination
1,0000 0,0000 0,0000 0,0000 -1,0000 3,0000
0,0000 1,0000 0,0000 0,0000 -2,0000 0,0000
0,0000 0,0000 0,0000 1,0000 -1,0000 -1,0000
0,0000 0,0000 0,0000 0,0000 0,0000 0,0000

Solusi banyak (parametrik):
x1 = 3,000 + 1,000x5
x2 = 2,000x5
x3 = c
x4 = -1,000 + 1,000x5
x5 = e
```

Jika disederhanakan, maka:
x1 = 3,000 + 1,000e
x2 = 2,000e

	x3 = c x4 = -1,000 + e x5 = e 3. Hasil Menggunakan Matriks Balikan Persamaan tidak dapat diselesaikan dengan metode invers SPL karena matrix koefisien bukan matrix persegi sehingga invers tidak dapat ditentukan. 4. Hasil Menggunakan Kaidah Cramer SPL tidak dapat diselesaikan dengan metode cramer.
1c	$\boldsymbol{A} = \begin{bmatrix} 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}, \quad \boldsymbol{b} = \begin{bmatrix} 2 \\ -1 \\ 1 \end{bmatrix}$
	1. Hasil Menggunakan Gauss Elimination Hasil Matrix Setelah Gauss Elimination: 0,000 1,000 0,000 0,000 1,000 0,000 2,000 0,000 1,000 0,000 0,000 0,000 1,000 1,000 0,000 0,000 0,000 1,000 1,000 0,000 -1,000 Solusi banyak (parametrik): x0 = a x1 = 1,000 - 1,000x6 x1 = c x2 = -1,000 - 1,000x5 x3 = c x4 = d Analisis : Kesalahan terjadi pada *swaprow* dan ketika seluruh elemen di suatu kolom bernilai nol selama eliminasi, maka algoritma seharusnya mengidentifikasi bahwa kolom tersebut tidak bisa dijadikan pivot. 2. Hasil Menggunakan Gauss-Jordan Elimination

```
Hasil Matrix Setelah Gauss Jordan Elimination:
0,000 0,000 0,000 0,000 1,000 -1,000 1,000
0,000 1,000 0,000 0,000 0,000 1,000 1,000
0,000 0,000 0,000 1,000 1,000 0,000 -1,000


Solusi banyak (parametrik):
x0 = a
x1 = 1,000 - 1,000x6
x1 = c
x2 = -1,000 - 1,000x5
x3 = c
x4 = d
```

Analisis : Kesalahan yang sama terjadi pada kasus gauss diatas.

3. Hasil Menggunakan Matriks Balikan

```
Masukkan nama file: test1c.txt
test/input/test1c.txt
Persamaan tidak dapat diselesaikan dengan metode invers SPL karena matrix koefisien bukan matrix
persegi sehingga invers tidak dapat ditentukan.
```

Analisis : Sudah Benar karena bukan matriks persegi

4. Hasil Menggunakan Kaidah Cramer

```
Masukkan nama file: test1c.txt
test/input/test1c.txt
SPL tidak dapat diselesaikan dengan metode cramer.
```

Analisis : Sudah benar karena bukan matriks persegi.

1d

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

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

n = 6

1. Hasil Menggunakan Gauss Elimination

```
Masukan jumlah ordo : 6
Hasil Matrix Setelah Gauss Elimination
1,0000 0,5000 0,3333 0,2500 0,2000 0,1667 1,0000
0,0000 1,0000 1,0000 0,9000 0,8000 0,7143 -6,0000
0,0000 0,0000 1,0000 1,5000 1,7143 1,7857 30,0000
0,0000 0,0000 0,0000 1,0000 2,0000 2,7778 -140,0000
0,0000 0,0000 0,0000 0,0000 1,0000 2,5000 630,0000
0,0000 0,0000 0,0000 0,0000 0,0000 1,0000 -2772,0000

Solusi tunggal:
X[1] = 36,0000
X[2] = -630,0000
X[3] = 3360,0000
X[4] = -7560,0000
X[5] = 7560,0000
X[6] = -2772,0000
```

2. Hasil Menggunakan Gauss Jordan Elimination

```
Masukan jumlah ordo : 6
Hasil Matrix Setelah Gauss Jordan Elimination
1,0000 0,0000 0,0000 0,0000 0,0000 0,0000 36,0000
0,0000 1,0000 0,0000 0,0000 0,0000 0,0000 -630,0000
0,0000 0,0000 1,0000 0,0000 0,0000 0,0000 3360,0000
0,0000 0,0000 0,0000 1,0000 0,0000 0,0000 -7560,0000
0,0000 0,0000 0,0000 0,0000 1,0000 0,0000 7560,0000
0,0000 0,0000 0,0000 0,0000 0,0000 1,0000 -2772,0000

Solusi tunggal:
X[1] = 36,0000
X[2] = -630,0000
X[3] = 3360,0000
X[4] = -7560,0000
X[5] = 7560,0000
X[6] = -2772,0000
```

3. Hasil Menggunakan Matriks Balikan

```
Masukan jumlah ordo : 6
X1 = 36,0000
X2 = -629,9999
X3 = 3359,9997
X4 = -7559,9994
X5 = 7559,9995
X6 = -2771,9999
```

4. Hasil Menggunakan Kaidah Cramer

```
Masukan jumlah ordo : 6
X1 = 36,0000
X2 = -629,9999
X3 = 3359,9997
X4 = -7559,9994
X5 = 7559,9995
X6 = -2771,9999
```

n = 10

1. Hasil Menggunakan Gauss Elimination

```
Masukan jumlah ordo : 10
Hasil Matrix Setelah Gauss Elimination
1,0000 0,5000 0,3333 0,2500 0,2000 0,1667 0,1429 0,1250 0,1111 0,1000 1,0000
0,0000 1,0000 1,0000 0,9000 0,8000 0,7143 0,6429 0,5833 0,5333 0,4909 -6,0000
0,0000 0,0000 1,0000 1,5000 1,7143 1,7857 1,7857 1,7500 1,6970 1,6364 30,0000
0,0000 0,0000 0,0000 1,0000 2,0000 2,7778 3,3333 3,7121 3,9596 4,1119 -140,0000
0,0000 0,0000 0,0000 0,0000 1,0000 2,5000 4,0909 5,5682 6,8531 7,9301 630,0000
0,0000 0,0000 0,0000 0,0000 0,0000 1,0000 3,0000 5,6538 8,6154 11,6308 -2772,0000
0,0000 0,0000 0,0000 0,0000 0,0000 0,0000 1,0000 3,5000 7,4667 12,6000 12012,0001
0,0000 0,0000 0,0000 0,0000 0,0000 0,0000 0,0000 1,0000 4,0000 9,5294 -51480,0012
0,0000 0,0000 0,0000 0,0000 0,0000 0,0000 0,0000 0,0000 1,0000 4,5000 218789,1295
0,0000 0,0000 0,0000 0,0000 0,0000 0,0000 0,0000 0,0000 0,0000 1,0000 -923630,2350

Solusi tunggal:
X[1] = 99,9963
X[2] = -4949,6827
X[3] = 79193,2148
X[4] = -600538,1369
X[5] = 2522224,2587
X[6] = -6305485,5472
X[7] = 9608261,7832
X[8] = -8750305,2142
X[9] = 4375119,5655
X[10] = -923630,2350
```

2. Hasil Menggunakan Gauss Jordan Elimination

```
Masukan jumlah ordo : 10
Hasil Matrix Setelah Gauss Jordan Elimination
1,0000 0,0000 0,0000 0,0000 0,0000 0,0000 0,0000 0,0000 0,0000 0,0000 99,9963
0,0000 1,0000 0,0000 0,0000 0,0000 0,0000 0,0000 0,0000 0,0000 0,0000 -4949,6827
0,0000 0,0000 1,0000 0,0000 0,0000 0,0000 0,0000 0,0000 0,0000 0,0000 79193,2148
0,0000 0,0000 0,0000 1,0000 0,0000 0,0000 0,0000 0,0000 0,0000 0,0000 -600538,1369
0,0000 0,0000 0,0000 0,0000 1,0000 0,0000 0,0000 0,0000 0,0000 0,0000 2522224,2587
0,0000 0,0000 0,0000 0,0000 0,0000 1,0000 0,0000 0,0000 0,0000 0,0000 -6305485,5472
0,0000 0,0000 0,0000 0,0000 0,0000 0,0000 1,0000 0,0000 0,0000 0,0000 9608261,7832
0,0000 0,0000 0,0000 0,0000 0,0000 0,0000 0,0000 1,0000 0,0000 0,0000 -8750305,2142
0,0000 0,0000 0,0000 0,0000 0,0000 0,0000 0,0000 0,0000 1,0000 0,0000 4375119,5655
0,0000 0,0000 0,0000 0,0000 0,0000 0,0000 0,0000 0,0000 0,0000 1,0000 -923630,2350

Solusi tunggal:
X[1] = 99,9963
X[2] = -4949,6827
X[3] = 79193,2148
X[4] = -600538,1369
X[5] = 2522224,2587
X[6] = -6305485,5472
X[7] = 9608261,7832
X[8] = -8750305,2142
X[9] = 4375119,5655
X[10] = -923630,2350
```

3. Hasil menggunakan Matriks Balikan

```
Masukan jumlah ordo : 10
X1 = 34,5902
X2 = -381,5402
X3 = 0,3756
X4 = 5698,2430
X5 = -9412,0433
X6 = -3176,6756
X7 = 7577,6072
X8 = 5393,8574
X9 = -2806,7101
X10 = -3005,4222
```

Analisis : Hasil untuk metode Matriks Balikan dengan masukan matriks hilbert orde 10 tidak sesuai dengan yang seharusnya, hal ini disebabkan karena terjadinya *overflow/underflow* dimana nilai melebihi kapasitas maksimum/minimum tipe data yang digunakan, sehingga ada nilai yang terlalu kecil dan dianggap 0. Pada implementasi penulis, hal ini berlaku untuk kode hilbert dengan ordo >= 8.

4. Hasil Menggunakan Kaidah cramer

```
Masukan jumlah ordo : 10
X1 = 34,5902
X2 = -381,5402
X3 = 0,3756
X4 = 5698,2430
X5 = -9412,0433
X6 = -3176,6756
X7 = 7577,6072
X8 = 5393,8574
X9 = -2806,7101
X10 = -3005,4222
```

Analisis : Permasalahan yang sama dengan yang terjadi pada metode Matriks Balikan.

2a

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

1. Hasil Menggunakan Gauss Elimination

```
Hasil Matrix Setelah Gauss Elimination
1.0000 -1.0000 2.0000 -1.0000 -1.0000
0.0000 1.0000 -2.0000 0.0000 0.0000
0.0000 0.0000 0.0000 0.0000 0.0000
0.0000 0.0000 0.0000 0.0000 0.0000

Solusi banyak (parametrik):
x1 = -1.000 + 1.000x2 - 2.000x3 + 1.000x4
x2 = 2.000x3
x3 = c
x4 = d
```

Jika disederhanakan menjadi

x1 = -1 + d
x2 = 2c
x3 = c
x4 = d

2. Hasil Menggunakan Gauss-Jordan Elimination

```
Hasil Matrix Setelah Gauss Jordan Elimination
1.0000 0.0000 0.0000 -1.0000 -1.0000
0.0000 1.0000 -2.0000 0.0000 0.0000
0.0000 0.0000 0.0000 0.0000 0.0000
0.0000 0.0000 0.0000 0.0000 0.0000

Solusi banyak (parametrik):
x1 = -1.000 + 1.000x4
x2 = 2.000x3
x3 = c
x4 = d
```

3. Hasil Menggunakan Matriks Balikan

```
SPL tidak dapat diselesaikan dengan matriks balikan karena nilai
balikan matriks tidak terdefinisi.
```

4. Hasil Menggunakan Kaidah Cramer

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

2b

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

1. Hasil Menggunakan Gauss Elimination

```
Hasil Matrix Setelah Gauss Elimination
1,0000 0,0000 4,0000 0,0000 4,0000
0,0000 1,0000 0,0000 4,0000 6,0000
0,0000 0,0000 1,0000 0,0000 1,0000
0,0000 0,0000 0,0000 1,0000 1,0000
0,0000 0,0000 0,0000 0,0000 0,0000
0,0000 0,0000 0,0000 0,0000 0,0000

Solusi banyak (parametrik):
x1 = 4,000 - 4,000x3
x2 = 6,000 - 4,000x4
x3 = 1,000
x4 = 1,000
```

Program mengeluarkan output solusi banyak karena mendeteksi adanya baris yang semuanya 0. Tetapi, karena hanya ada 4 variabel dan 4 persamaan selain baris yang bernilai 0, sehingga solusi dapat ditentukan sesuai dengan output yang tertera pada program

x1 = 0,000
x2 = 2,000
x3 = 1,000
x4 = 1,000

2. Hasil Menggunakan Gauss Jordan Elimination

```
Hasil Matrix Setelah Gauss Jordan Elimination
1,0000 0,0000 0,0000 0,0000 0,0000
0,0000 1,0000 0,0000 0,0000 2,0000
0,0000 0,0000 1,0000 0,0000 1,0000
0,0000 0,0000 0,0000 1,0000 1,0000
0,0000 0,0000 0,0000 0,0000 0,0000
0,0000 0,0000 0,0000 0,0000 0,0000

Solusi banyak (parametrik):
x1 =
x2 = 2,000
x3 = 1,000
x4 = 1,000
```

Penjelasan sama dengan metode Gauss.

3. Hasil Menggunakan Matriks Balikan

```
Persamaan tidak dapat diselesaikan dengan metode invers SPL karena matrix koefisien bukan
matrix persegi sehingga invers tidak dapat ditentukan.
```

4. Hasil Menggunakan Kaidah Cramer

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

3a

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

1. Hasil Menggunakan Gauss Elimination

```
Hasil Matrix Setelah Gauss Elimination
1.0000 0.1250 0.3750 0.2500 0.0000
0.0000 1.0000 -0.2000 -0.2857 0.1143
0.0000 0.0000 1.0000 -0.1948 0.7597
0.0000 0.0000 0.0000 1.0000 -0.2581

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

2. Hasil Menggunakan Gauss-Jordan Elimination

```
Hasil Matrix Setelah Gauss Jordan Elimination
1.0000 0.0000 0.0000 0.0000 -0.2243
0.0000 1.0000 0.0000 0.0000 0.1824
0.0000 0.0000 1.0000 0.0000 0.7095
0.0000 0.0000 0.0000 1.0000 -0.2581

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

3. Hasil Menggunakan Matriks Balikan

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

4. Hasil Menggunakan Kaidah Cramer

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

3b

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

1. Hasil Menggunakan Gauss Elimination

```
1.0000 1.0000 1.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 8.0000
0.0000 1.0000 3.6568 1.0000 3.6568 1.0000 3.6568 1.0000 0.0000 57.2400
0.0000 0.0000 1.0000 0.0000 1.0000 17.4866 1.0000 17.4866 14.3148 344.8356
0.0000 0.0000 0.0000 1.0000 1.0000 1.0000 0.0000 0.0000 0.0000 15.0000
0.0000 0.0000 0.0000 0.0000 1.0000 -161546268097193380.0000 -5530060054440778.0000 -167076328151634140.0000 -137773989069044560.0000 -3155896505272857000.0000
0.0000 0.0000 0.0000 0.0000 0.0000 1.0000 0.0342 1.0342 0.8528 19.5356
0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 1.0000 1.0000 1.0000 13.0000
0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 1.0000 17.0000 13.0000
0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 1.0000 -16.0000
0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 1.0000
0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000

Solusi tidak ada.
```

2. Hasil Menggunakan Gauss-Jordan Elimination

```
Hasil Matrix Setelah Gauss Jordan Elimination
1.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
0.0000 1.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
0.0000 0.0000 1.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
0.0000 0.0000 0.0000 1.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
0.0000 0.0000 0.0000 0.0000 1.0000 0.0000 0.0000 0.0000 0.0000 0.0000
0.0000 0.0000 0.0000 0.0000 0.0000 1.0000 0.0000 0.0000 0.0000 0.0000
0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 1.0000 0.0000 0.0000 0.0000
0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 1.0000 0.0000 0.0000
0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 1.0000 0.0000
0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 1.0000
0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000

Solusi tidak ada.
```

3. Hasil Menggunakan Matriks Balikan

```
Persamaan tidak dapat diselesaikan dengan metode invers SPL karena
 matrix koefisien bukan matrix persegi sehingga invers tidak dapat
 ditentukan.
```

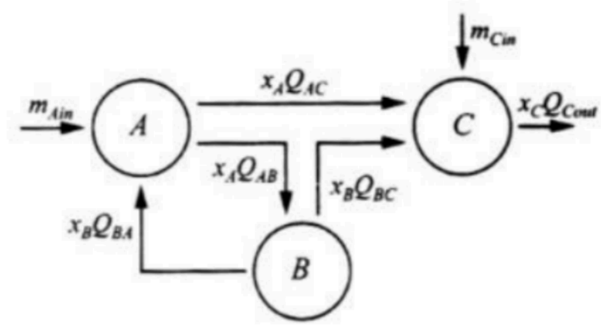
4. Hasil Menggunakan Kaidah Cramer

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

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

</td></tr>
<tr><td>4</td><td>

4. Lihatlah sistem reaktor pada gambar berikut.



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

$$
\begin{aligned}
&\text{A:} \quad m_{A_{in}} + Q_{BA}x_B - Q_{AB}x_A - Q_{AC}x_A = 0 \\
&\text{B:} \quad Q_{AB}x_A - Q_{BA}x_B - Q_{BC}x_B = 0 \\
&\text{C:} \quad m_{C_{in}} + Q_{AC}x_A + Q_{BC}x_B - Q_{C_{out}}x_C = 0
\end{aligned}
$$

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.

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

1. Hasil Menggunakan Gauss Elimination

```
Hasil Matrix Setelah Gauss Elimination
1,0000 -0,5000 -0,0000 10,8333
0,0000 1,0000 -0,0000 7,2222
0,0000 0,0000 1,0000 10,0000

Solusi tunggal:
X[1] = 14,4444
X[2] = 7,2222
X[3] = 10,0000
```

2. Hasil Menggunakan Gauss-Jordan Elimination

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

```
Hasil Matrix Setelah Gauss Jordan Elimination
1,0000 0,0000 0,0000 14,4444
0,0000 1,0000 0,0000 7,2222
0,0000 0,0000 1,0000 10,0000

Solusi tunggal:
X[1] = 14,4444
X[2] = 7,2222
X[3] = 10,0000
```

3. Hasil Menggunakan Matriks Balikan

```
X1 = 14,4444
X2 = 7,2222
X3 = 10,0000
```

4. Hasil Menggunakan Kaidah Cramer

```
X1 = 14,4444
X2 = 7,2222
X3 = 10,0000
```

5a

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

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

Lakukan pengujian pada nilai-nilai berikut:

x = 0.2 *f*(*x*) = ?

x = 0.55 *f*(*x*) = ?

x = 0.85 *f*(*x*) = ?

x = 1.28 *f*(*x*) = ?

x = 0.2

```
f(x) = - 0,0000x^6 + 0,0000x^5 + 0,0260x^4 + 0,0000x^3 + 0,1974x^2
+ 0,2400x - 0,0230,
f(0,2000) = 0,0330
```

x = 0.55

```
f(x) = - 0,0000x^6 + 0,0000x^5 + 0,0260x^4 + 0,0000x^3 + 0,1974x^2
 + 0,2400x - 0,0230,
f(0,5500) = 0,1711
```

x = 0.85

```
f(x) = - 0,0000x^6 + 0,0000x^5 + 0,0260x^4 + 0,0000x^3 + 0,1974x^2
 + 0,2400x - 0,0230,
f(0,8500) = 0,3372
```

x = 1.28

```
f(x) = - 0,0000x^6 + 0,0000x^5 + 0,0260x^4 + 0,0000x^3 + 0,1974x^2
 + 0,2400x - 0,0230,
f(1,2800) = 0,6775
```

5b

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

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

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

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.

a. 16/07/2022

```
f(x) = - 141,0191x^9 + 9374,6355x^8 - 275530,2683x^7 + 4696819,2596x^6 - 51143697,7386x^5
 + 368642656,6894x^4 - 1757285362,0195x^3 + 5335781355,2109x^2 - 9350047182,6158x + 71896
89299,2108,
f(7,5160) = 53,5372
```

b. 10/08/2022

```
f(x) = - 141,0191x^9 + 9374,6355x^8 - 275530,2683x^7 + 4696819,2596x^6 - 51143697,7386x^5
 + 368642656,6894x^4 - 1757285362,0195x^3 + 5335781355,2109x^2 - 9350047182,6158x + 71896
89299,2108,
f(8,3230) = 36,2947
```

c. 05/09/2022

```
f(x) = - 141,0191x^9 + 9374,6355x^8 - 275530,2683x^7 + 4696819,2596x^6 - 51143697,7386x^5
 + 368642656,6894x^4 - 1757285362,0195x^3 + 5335781355,2109x^2 - 9350047182,6158x + 71896
89299,2108,
f(9,1670) = -667,7292
```

d. f(x) dengan x sesuai pengguna (Disini penulis memasukkan x=10 ke dalam file sehingga menghasilkan output sebagai berikut)

```
f(x) = - 141,0191x^9 + 9374,6355x^8 - 275530,2683x^7 + 4696819,2596x^6 - 51143697,7386x^5
 + 368642656,6894x^4 - 1757285362,0195x^3 + 5335781355,2109x^2 - 9350047182,6158x + 71896
89299,2108,
f(10,0000) = -216947,0577
```

5c

Sederhanakan fungsi $f(x)$ yang memenuhi kondisi

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

dengan polinom interpolasi derajat n di dalam selang [0, 2].

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

```
f(x) = 0,0078x^3 - 0,1250x^2 + 0,3612x + 0,2940,
```

6

Studi Kasus Regresi Linear dan Kuadratik Berganda

Diberikan sekumpulan data sesuai pada tabel berikut ini.

Table 12.1: Data for Example 12.1

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

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

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

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

$$20b_0 + 863.1b_1 + 1530.4b_2 + 587.84b_3 = 19.42$$

$$863.1b_0 + 54876.89b_1 + 67000.09b_2 + 25283.395b_3 = 779.477$$

$$1530.4b_0 + 67000.09b_1 + 117912.32b_2 + 44976.867b_3 = 1483.437$$

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

$587.84b_0 + 25283.395b_1 + 44976.867b_2 + 17278.5086b_3 = 571.1219$

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

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

Regresi Linear Berganda

```
f(x) = - 3,5078 - 0,0026x1 + 0,0008x2 + 0,1542x3,
f(xk) = 0,9384
```

Regresi Kuadratik Berganda

```
f(x) = - 866,7091 - 0,0035x1 + 0,0079x2 + 58,8817x3 - 0,2541x4
 + 0,0000x5 - 0,0001x6^2,
f(xk) = 846,2389
```

Analisis : Untuk Regresi Kuadratik Berganda yang penulis implementasikan, sepertinya masih terdapat kesalahan sehingga hasil jauh dari yang diharapkan. Seharusnya, nilai estimasi pada Regresi Linier Berganda sudah sesuai karena dalam program yang penulis buat, data akan diubah ke dalam bentuk *Normal Estimation Equation for Multiple Linear Regression* dan diselesaikan dengan pendekatan matriks metode gauss.

</td></tr>
<tr><td>7</td><td>

Studi Kasus Interpolasi Bicubic Spline

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

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

Tentukan nilai:

$f(0, 0) = ?$

$f(0.5, 0.5) = ?$

$f(0.25, 0.75) = ?$

$f(0.1, 0.9) = ?$

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

1. $f(0, 0)$

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

	`f(0.0,0.0) = 21.0000`
	2. f(0.5, 0.5)
	`f(0.5,0.5) = 87.7969`
	3. f(0.25, 0.75)
	`f(0.25,0.75) = 82.1482`
	4. f(0.1, 0.9)
	`f(0.1,0.9) = 91.2713`

BAB V
KESIMPULAN, SARAN, KOMENTAR DAN REFLEKSI

5.1 Kesimpulan

Sistem Persamaan Linear (SPL) merupakan kumpulan persamaan yang mengandung variabel-variabel, di mana tujuan utamanya adalah menemukan solusi yang memenuhi semua persamaan dalam sistem tersebut. Solusi ini merepresentasikan titik di ruang multidimensi yang sesuai dengan setiap hubungan dalam sistem. Ada beberapa topik penting dalam SPL, termasuk mencari solusi SPL, determinan, matriks invers, interpolasi polinomial, interpolasi bicubic spline, dan regresi linier berganda. Dalam penyelesaian SPL, beberapa metode yang digunakan adalah metode eliminasi Gauss, eliminasi Gauss-Jordan, metode matriks invers, dan aturan Cramer.

Untuk mencapai solusi dalam SPL, penulis menggunakan konsep matriks, yaitu kumpulan array dalam program, serta metode-metode yang telah disebutkan. Solusi SPL dapat ditemukan menggunakan salah satu dari empat metode tersebut. Untuk menghitung determinan, metode eliminasi Gauss (mengubah matriks menjadi bentuk segitiga atas atau bawah) dan metode kofaktor dapat diterapkan. Untuk mendapatkan matriks invers, metode eliminasi Gauss-Jordan dan metode kofaktor juga dapat digunakan, dengan syarat bahwa determinan matriks tidak boleh bernilai nol dan matriks harus berbentuk persegi.

Dengan konsep-konsep dasar SPL ini, berbagai program dapat dikembangkan, seperti interpolasi polinomial yang digunakan untuk memperkirakan nilai di antara data yang ada. Interpolasi bicubic spline menghasilkan fungsi interpolasi yang halus dan berkelanjutan melalui titik-titik data yang diberikan. Ditunjukkan juga pada tugas besar ini, konsep-konsep yang ada bisa mengubah ukuran suatu gambar terhadap skala tertentu. Sementara itu, regresi linier berganda berguna untuk menganalisis hubungan antara satu atau lebih variabel independen (prediktor) dengan satu variabel dependen (yang diprediksi).

5.2 Saran

Saran yang diharapkan untuk tugas besar ini adalah untuk setiap test case yang ada disertai juga dengan hasil dari test case tersebut. Dengan mengetahui hasil yang ada, penulis dapat mengetahui apakah kode yang dibuat sudah benar atau belum.

5.3 Komentar

INTERUPSI ARES! Maaf masih ada program yang outputnya tidak sesuai. We love you Ares <3 kata soni jangan galak galak pls :(

5.4 Refleksi

Pengerjaan tugas besar 1 Aljabar Linier dan Geometri membuat penulis lebih memahami terkait hal-hal seputar Sistem Persamaan Linier dan hal lain yang berhubungan dengan itu. Dalam pengerjaan tugas besar ini, diperlukan ketelitian dalam pembacaan spek sehingga tidak salah dalam implementasinya. Selain itu, perlu dieksplor lebih banyak kasus, sehingga semua kondisi dapat ditemukan penyelesaian yang tepat. Penulis juga mendapatkan peningkatan dalam beberapa kemampuan, seperti kemampuan bekerja sama, kemampuan manajemen waktu, dan kemampuan pemrograman dengan bahasa yang baru penulis pelajari.

LAMPIRAN

1. Blog Demofox. 2015. “Resizing Image with Bicubic Interpolation” https://blog.demofox.org/2015/08/15/resizing-images-with-bicubic-interpolation/
2. Codingtechroom.”Understanding setRGB() Method in Java: A Comprehensive Guide” https://codingtechroom.com/question/understanding-setrgb-method-in-java-a-comprehensive-guide
3. Java Code Geeks. 2013. “Get and Set Pixels on Buffered Image” https://examples.javacodegeeks.com/java-development/desktop-java/awt/image/get-and-set-pixels-on-a-buffered-image/
4. Munir, Rinaldi. 2023. “Sistem persamaan linear (Bagian 1: Metode Eliminasi Gauss)” https://informatika.stei.itb.ac.id/~rinaldi.munir/AljabarGeometri/2023-2024/Algeo-03-Sistem-Persamaan-Linier-2023.pdf
5. Munir, Rinaldi. 2023. “Sistem persamaan linear (Bagian 2 : Tiga Kemungkinan Solusi Sistem Persamaan Linier)” https://informatika.stei.itb.ac.id/~rinaldi.munir/AljabarGeometri/2023-2024/Algeo-04-Tiga-Kemungkinan-Solusi-SPL-2023.pdf
6. Munir, Rinaldi. 2023. “Sistem persamaan linear (Bagian 2 : Metode Eliminasi Gauss Jordan)” https://informatika.stei.itb.ac.id/~rinaldi.munir/AljabarGeometri/2023-2024/Algeo-05-Sistem-Persamaan-Linier-2-2023.pdf
7. Munir, Rinaldi. 2023. “Aplikasi Metode Eliminasi Gauss di dalam Metode Numerik” https://informatika.stei.itb.ac.id/~rinaldi.munir/AljabarGeometri/2023-2024/Algeo-07-Aplikasi-SPL-2-2023.pdf

Link Repository : https://github.com/fithrarzk/Algeo01-23049

Link Video : https://youtu.be/hvT9VmouNU8?si=OW9vj58Et7TSRkXO