Tugas Besar I

IF 2123 Aljabar Linear dan Geometri

Sistem Persamaan Linier, Determinan, dan Aplikasinya

Semester I Tahun 2023/2024

Kelompok Apick :

1. **Keanu Amadius G. W. 13522082**
2. **Abdullah Mubarak 13522101**
3. **Dimas Bagoes H. 13522112**

PROGRAM STUDI TEKNIK INFORMATIKA

SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA INSTITUT TEKNOLOGI BANDUNG 2023

DAFTAR ISI

BAB I

Deskripsi Masalah

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

Di dalam Tugas Besar 1 ini, telah dibuat 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.

BAB II

Teori Singkat

1. Eliminasi Gauss

Matriks segitiga atas yang didapat dari algoritma ini akan memiliki bentuk eselon baris (*row echelon form*). Jika semua koefisien utama (nilai bukan nol pertama pada sebuah baris) matriks bernilai 1, dan kolom-kolom yang mengandung koefisien utama memiliki bentuk yang sama dengan kolom pada matriks identitas, matriks tersebut dikatakan memiliki bentuk eselon baris tereduksi (*reduced row echelon form*). Eliminasi Gauss yang dilakukan untuk mengubah matriks koefisien *sampai* menjadi bentuk eselon baris tereduksi terkadang disebut sebagai eliminasi Gauss–Jordan. Karena alasan komputasi, operasi baris untuk mencari solusi sistem persamaan terkadang dihentikan sebelum matriks berada dalam bentuk tereduksinya. Misalkan kita ingin menemukan dan mendeskripsikan himpunan solusi ke sistem persamaan linear berikut:

$$\begin{aligned} 2x + y - z &= 8 && (L_1) \\ -3x - y + 2z &= -11 && (L_2) \\ -2x + y + 2z &= -3 && (L_3) \end{aligned}$$

Gambar 2.1 Contoh SPL

Persamaan tersebut dapat dinyatakan dalam bentuk matriks koefisien gabungan:

$$\left[\begin{array}{ccc|c} 2 & 1 & -1 & 8 \\ -3 & -1 & 2 & -11 \\ -2 & 1 & 2 & -3 \end{array}\right]$$

Gambar 2.2 Matriks hasil konversi dari SPL pada Gambar 2.1

Setelah menyusun sistem persamaan linear menjadi sebuah matriks koefisien, eliminasi Gauss melakukan serangkaian operasi baris dasar untuk “menyederhanakan” baris-baris matriks. Eliminasi ini dapat dibagi menjadi dua bagian. Bagian pertama, terkadang disebut dengan eliminasi maju, mereduksi sistem yang diberikan ke dalam bentuk *eselon baris*. Berikut contoh matriks 4x5 berbentuk eselon baris, yang tidak berbentuk eselon baris tereduksi

$$\begin{bmatrix} 1 & a_0 & a_1 & a_2 & a_3 \\ 0 & 0 & 2 & a_4 & a_5 \\ 0 & 0 & 0 & 1 & a_6 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Gambar 2.3 Contoh matriks eselon barisambar

Ada tiga jenis operasi baris dasar yang dapat dilakukan pada baris matriks tersebut:

1. Menukar posisi dua baris.
2. Mengalikan suatu baris dengan skalar bukan nol .

3. Menambahkan suatu baris dengan suatu kelipatan dari baris yang lain.

2. Eliminasi Gauss-Jordan

Eliminasi Gauss Jordan adalah suatu metode menyelesaikan sistem persamaan linear dengan membuat matriks ke bentuk matriks eselon baris tereduksi. Matriks eselon baris tereduksi memiliki karakteristik yang sama dengan matriks eselon baris dengan satu tambahan karakteristik, yaitu seluruh kolom yang mengandung leading one tidak memiliki angka lain selain 0 di kolom tersebut. Hasil dari bagian ini dapat memberitahu apakah sistem persamaan linear tidak memiliki solusi, memiliki solusi unik, atau memiliki tak hingga solusi. Bagian kedua, terkadang disebut substitusi mundur, melanjutkan penggunaan operasi baris dasar sampai matriks berada dalam bentuk *eselon baris tereduksi*, dengan kata lain, hingga solusi ditemukan. Berikut adalah contoh matriks berbentuk eselon baris tereduksi

$$\begin{bmatrix} 1 & 0 & a_1 & 0 & b_1 \\ 0 & 1 & a_2 & 0 & b_2 \\ 0 & 0 & 0 & 1 & b_3 \end{bmatrix}$$

Gambar 2.4 Contoh matriks esekon baris tereduksi

Sama halnya seperti pada Eliminasi Gauss, membuat matriks menjadi bentuk matriks eselon baris tereduksi dapat dilakukan dengan memanfaatkan Operasi Baris Elementer (OBE). Dalam melakukan OBE pada Eliminasi Gauss Jordan, algoritma dapat dibagi menjadi dua bagian, yaitu:

1. Forward phase: tahap membuat 0 di bawah leading one, dengan kata lain membuat bentuk matriks eselon baris.
2. Backward phase: tahap membuat 0 di atas leading one yang menghasilkan matriks eselon baris tereduksi.

Setelah memperoleh matriks eselon baris tereduksi melalui Operasi Baris Elementer, diperoleh beberapa persamaan. Dari persamaan-persamaan yang diperoleh, dapat dicari solusi dari sistem persamaan linear

3. Determinan

Dalam bidang aljabar linear, determinan adalah nilai yang dapat dihitung dari unsur suatu matriks persegi. Determinan matriks *A* ditulis dengan tanda det(*A*), det *A*, atau |*A*|. Determinan dapat dianggap sebagai faktor penskalaan transformasi yang

digambarkan oleh matriks. Untuk setiap matriks persegi, kita dapat mengasosiasikan suatu angka yang merupakan determinan matriks tersebut. Matriks persegi sendiri merupakan suatu matriks yang memiliki jumlah kolom dan baris yang sama. Determinan dari A dilambangkan dengan |A|, atau dapat dilambangkan secara langsung dalam istilah entri matriks dengan menulis batang penutup, bukan tanda kurung:

$$\begin{vmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,n} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n,1} & a_{n,2} & \cdots & a_{n,n} \end{vmatrix}.$$

Gambar 2.5 Lambang determinan matriks

Apabila matriksnya berbetuk 2 × 2, rumus untuk mencari determinan adalah:

$$|A| = \begin{vmatrix} a & b \\ c & d \end{vmatrix} = ad - bc.$$

Gambar 2.6 Determinan matriks 2 × 2

Apabila matriksnya berbentuk 3 × 3 matrix A, rumusnya adalah:

$$|A| = \begin{vmatrix} a & b & c \\ d & e & f \\ g & h & i \end{vmatrix} = a \begin{vmatrix} e & f \\ h & i \end{vmatrix} - b \begin{vmatrix} d & f \\ g & i \end{vmatrix} + c \begin{vmatrix} d & e \\ g & h \end{vmatrix}$$
$$= aei + bfg + cdh - ceg - bdi - afh.$$

Gambar 2.7 Determinan matriks 3 × 3

4. **Matriks Balikan**

Matriks juga memiliki balikan yang disebut matriks invers. Namun demikian, tidak semua matriks akan memiliki invers matriks. Agar sebuah matriks memiliki invers, maka matriks tersebut harus berupa matriks persegi. Selain itu, sebuah matriks yang memiliki invers harus memiliki determinan yang tidak boleh sama dengan nol.

Jika sebuah matriks memiliki invers, maka matriks tersebut disebut invertibel atau dapat diinversi. Sedangkan jika sebuah matriks tidak dapat diinversi, maka matriks tersebut disebut matriks singular. Matriks balikan dapat dicari menggunakan 2 cara yaitu dengan adjoin atau menggunakan eliminasi Gauss Jordan. Rumus invers matriks dengan metode adjoin yaitu:

$$M^{-1} = \frac{1}{\det M} C^T$$

Gambar 2.8 Matriks Invers

Dengan M^{-1} menyatakan invers matriks, det M menyatakan determinan matriks yang dicari inversnya, dan CT adalah transpose matriks yang elemen-elemennya adalah kofaktor matriks M atau bisa disevut adjoin.

5. Matriks Kofaktor

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

$$Cij = (-1)^{i+j} Mij$$

Gambar 2.9 Matriks Kofaktor

Sama seperti minor jumlah kofaktor suatu matriks mengikuti jumlah elemen matriks tersebut. Untuk contoh saya akan melanjutkan contoh 1 dan contoh 2 yang minornya sudah ditentukan sebelumnya

6. Adjoin Matriks

Adjoin matriks merupakan transpose dari suatu matriks yang elemen-elemennya merupakan kofaktor dari elemen-elemen matriks tersebut. Adjoin dapat dihasilkan dari matriks lain dengan cara menggantikan setiap elemen matriks dengan kofaktor atau minor yang sesuai dari matriks tersebut. Adjoin dapat digunakan dalam menghitung suatu matriks.

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. 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.

Berikut adalah sistem persamaan linear:

$$\begin{cases} a_1x + b_1y &= c_1 \\ a_2x + b_2y &= c_2 \end{cases}$$

Gambar 2.10 SPL 2 Variabel

Matriks persamaan ini adalah:

$$\begin{bmatrix} a_1 & b_1 \\ a_2 & b_2 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} c_1 \\ c_2 \end{bmatrix}.$$

Gambar 2.11 SPL 2 Variabel Dalam Matriks

Apabila a1b2 − b1a2 bukan nol, maka x dan y dapat dicari dengan menggunakan determinan matriks tersebut:

$$x = \frac{\begin{vmatrix} c_1 & b_1 \\ c_2 & b_2 \end{vmatrix}}{\begin{vmatrix} a_1 & b_1 \\ a_2 & b_2 \end{vmatrix}} = \frac{c_1b_2 - b_1c_2}{a_1b_2 - b_1a_2}, \quad y = \frac{\begin{vmatrix} a_1 & c_1 \\ a_2 & c_2 \end{vmatrix}}{\begin{vmatrix} a_1 & b_1 \\ a_2 & b_2 \end{vmatrix}} = \frac{a_1c_2 - c_1a_2}{a_1b_2 - b_1a_2}.$$

Gambar 2.12 Kaidah Cramer Untuk SPL 2 Variabel

Untuk matriks 3 × 3, caranya sama:

$$\begin{cases} a_1x + b_1y + c_1z &= d_1 \\ a_2x + b_2y + c_2z &= d_2 \\ a_3x + b_3y + c_3z &= d_3 \end{cases}$$

Gambar 2.13 SPL 3 Variabel

ersamaan ini dalam bentuk matriks adalah sebagai berikut:

$$\begin{bmatrix} a_1 & b_1 & c_1 \\ a_2 & b_2 & c_2 \\ a_3 & b_3 & c_3 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} d_1 \\ d_2 \\ d_3 \end{bmatrix}.$$

Gambar 2.14 SPL 3 Variabel Dalam Matriks

Kemudian nilai x, y dan z dapat dicari dengan rumus berikut:

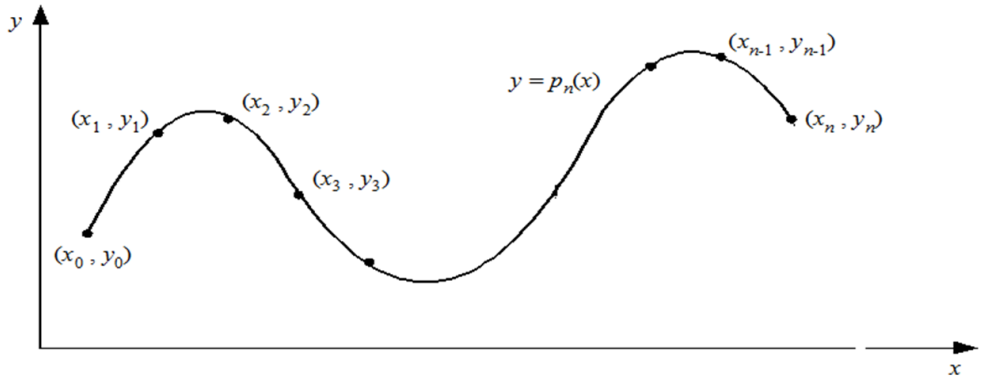
$$x = \frac{\begin{vmatrix} d_1 & b_1 & c_1 \\ d_2 & b_2 & c_2 \\ d_3 & b_3 & c_3 \end{vmatrix}}{\begin{vmatrix} a_1 & b_1 & c_1 \\ a_2 & b_2 & c_2 \\ a_3 & b_3 & c_3 \end{vmatrix}}, \quad y = \frac{\begin{vmatrix} a_1 & d_1 & c_1 \\ a_2 & d_2 & c_2 \\ a_3 & d_3 & c_3 \end{vmatrix}}{\begin{vmatrix} a_1 & b_1 & c_1 \\ a_2 & b_2 & c_2 \\ a_3 & b_3 & c_3 \end{vmatrix}}, \text{ dan } z = \frac{\begin{vmatrix} a_1 & b_1 & d_1 \\ a_2 & b_2 & d_2 \\ a_3 & b_3 & d_3 \end{vmatrix}}{\begin{vmatrix} a_1 & b_1 & c_1 \\ a_2 & b_2 & c_2 \\ a_3 & b_3 & c_3 \end{vmatrix}}.$$

Gambar 2.15 Kaidah Cramer Untuk SPL 3 Variabel

8. Interpolasi Polinom

Interpolasi Polinom merupakan suatu teknik untuk menentukan nilai yang tidak diketahui di antara beberapa nilai yang diketahui. 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 2.16 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, 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 $a_0, a_1, a_2, \ldots, a_n$,

$$a_0 + a_1x_0 + a_2x_0^2 + ... + a_n x_0^n = y_0$$

$$a_0 + a_1x_1 + a_2x_1^2 + ... + a_n x_1^n = y_1$$

$$... \qquad ...$$

$$a_0 + a_1x_n + a_2x_n^2 + ... + a_n x_n^n = y_n$$

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: $p_2(9.2) = 0.6762 + 0.2266(9.2) - 0.0064(9.2)^2 = 2.2192$.

9. Interpolasi *Bicubic Spline*

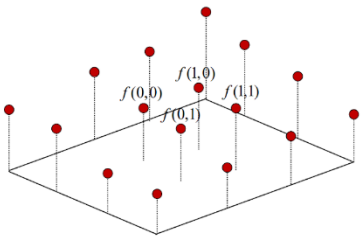
Bicubic spline interpolation merupakan metode penerapan interpolasi kubik pada kumpulan data untuk menginterpolasi titik data pada kisi reguler dua dimensi . Metode ini melibatkan konsep *spline* dan konstruksi serangkaian polinomial kubik di dalam setiap sel segi empat dari data yang diberikan.

Ada16 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 2.17. Pemodelan interpolasi *bicubic spline*.

Persamaan polinomial yang digunakan adalah model turunan berarah dari kedua sumbu, baik sumbu x, sumbu y ataupun keduanya.

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

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

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

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

Gambar 2.18 Turunan Berarah

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 2.19 Matriks Turunan Berarah

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.

10. Regresi Linear Berganda

Regresi linear berganda merupakan model regresi yang melibatkan lebih dari satu variabel independen. Analisis regresi linear berganda dilakukan untuk mengetahui arah dan seberapa besar pengaruh variabel independen terhadap variabel dependen.

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 2.20 Persamaan Umum RLB

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

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

Gambar 2.21 *Normal Estimation Equation for Multiple Linear Regression*

Sistem persamaan linier tersebut diselesaikan dengan menggunakan metode eliminasi Gauss.

BAB III

Implementasi

A. Implementasi Pustaka

1. Cramer.java

- Atribut : --
- Method :

a. public static String cramer(Matrix m1)

Mengembalikan string solusi persamaan linear yang didapatkan menggunakan kaidah Cramer. Jika matriks m1 bukan matriks persegi, maka string yang dikembalikan adalah “Kaidah Cramervtidak berlaku untuk input. Silakan gunakan metode lain.”, atau jika matriks tidak memiliki solusi unik, string yang dikembalikan adalah "SPL memiliki banyak solusi atau tidak memiliki solusi. Silakan gunakan metode lain.".

2. Determinan.java

- Atribut : --
- Method :

a. public static Matrix getMinor(Matrix m1,int row, int col)

Fungsi ini akan menghasilkan matriks minor tanpa row dan col tertentu. Fungsi ini akan mnghasilkan matriks dengan besar n-1 dari matriks masukan.

b. public static double determinanKofaktor(Matrix m1)

Fungsi ini akan menghasilkan nilai determinan dengan menggunakan metode kofaktor. Fungsi ini memanfaatkan fungsi getMinor. Fungsi ini dijalankan dengan mengambil baris pertama untuk kofaktornya. Sehingga hasil minor nya tanpa baris pertama.

c. public static double determinanGauss(Matrix m1)

Fungsi ini akan menghasilkan nilai determinan dengan menggunakan metode kofaktor dimana jika ada proses swap baris maka determinan akan dikali (-1). Jika suatu baris dikali n maka determinan akan dibagi dengan n.

3. Gauss.java

- Atribut :--
- Method :

a. public static double rounding(double n)

Fungsi ini akan membulatkan input n ke bilangan bulat terdekat jika selisih n dengan bilangan bulat terdekat kurang dari 10^{-8}.

b. public static Matrix add(Matrix n, int f, int s, double x)

Fungsi ini menambah setiap anggota row f dengan setiap anggota row s yang sudah dikali dengan x pada matriks n.

c. public static Matrix swap(Matrix n, int r_1, int r_2)

Fungsi ini menukar baris r_1 dengan baris r_2 pada matriks n.

d. public static Matrix multi(Matrix n, int row, double x)

Fungsi ini mengalikan setiap anggota baris row pada matriks n dengan nilai x.

e. public static void first(Matrix n, int r_start, int col)

Prosedur ini mengecek nilai pada n[r_start][col], jika nilainya nol maka baris r_start akan ditukar(swap) dengan baris yang elemen pada kolom col nya tidak nol.

f. public static boolean colZero(Matrix n, int row, int col)

Fungsi ini mencari kolom pada matriks n dengan indeks baris row sampai n.row-1 yang semua anggotanya bernilai nol.

g. public static Matrix eliminasiGauss(Matrix n, boolean x)

Fungsi ini mengembalika input matriks n menjadi matriks eselon baris. X bernilai true jika matriks n adalah matriks augmented dan bernilai false jika matriks n adalah matriks biasa.

4. Gauss_jordan.java

- Atribut :--
- Method :

a. public static int getFirstOne(Matrix n, int row)

Fungsi ini mengembalikan index kolom satu pertama pada baris row pada matriks n.

b. public static Matrix eliminasiGaussJordan

Fungsi ini mengembalika input matriks n menjadi matriks eselon baris tereduksi. X bernilai true jika matriks n adalah matriks augmented dan bernilai false jika matriks n adalah matriks biasa.

5. Invers.java

- Atribut : --
- Method :

a. public static Matrix inversAdjoint(Matrix m1)

Mengembalikan Matrix berupa matriks balikan dari Matrix m1 dengan matrix m1 harus berupa matriks persegi dengan determinan ≠ 0. Matriks balikan diperoleh melalui matriks adjoint.

b. public static Matrix inversGaussJordan(Matrix m1)

Mengembalikan Matrix berupa matriks balikan dari Matrix m1 dengan matrix m1 harus berupa matriks persegi. Matriks balikan diperoleh melalui metode Gauss Jordan.

6. Matrix.java

- Atribut :

public int row;

public int col;

public double[][] m;

private static Scanner scan = new Scanner(System.in);

- Method :

a. public Matrix(int row, int col)

membuat matrix dengan baris dan kolom

b. public void inputRowCol()

melakukan proses input baris dan kolom matrix

F.S. baris dan kolom terdefinisi.

c. public void readMatrix()

membaca matrix

I.S. Baris dan Kolom sudah terdefinisi.

d. public void displayMatrix()

Menampilkan matriks dengan type double 4 angka dibelakang koma

e. public Matrix createIdentitas()

Membuat matrix identitas

I.S. Baris dan Kolom harus sama dan sudah terdefinisi

f. public void multiplyCons(double k)

mengalikan matrix dengan konstanta k

g. public Matrix multiplyMatrix(Matrix m1, Matrix m2)

mengalikan matrix m1 dengan matrix m2.

h. public void copyMatrix(Matrix mIn)

mengopy matrix ke matrix mIn.

i. public Matrix addMatrix(Matrix m1, Matrix m2)

Menjumlahkan m1 dengan m2 (m1+m2)

I.S. ukuran m1 dan m2 harus sama

j. public Matrix substractMatrix(Matrix m1, Matrix m2)

Mengurangi m1 dengan m2 (m1-m2)

I.S. ukuran m1 dan m2 harus sama

k. public Matrix transpose()

menghasilkan transpose dari suatu matrix.

l. public boolean isSquare()

menghasilkan true apabila matrix merupakan square

m. public boolean isSymmetric()

menghasilkan true apabila matrix symetric

n. public boolean isIdentity()

menghasilkan true apabila matrix identitas.

o. public boolean isMatrixEqual(Matrix m1, Matrix m2)

menghasilkan true jika semuau elemen matrix m1 sama dengan matrix m2

p. public boolean isMatrixNotEqual(Matrix m1, Matrix m2)

menghasilkan true jika elemen matrix m1 tidak sama dengan matrix m2.

q. public boolean isMatrixSizeEqual(Matrix m1, Matrix m2)

menghasilkan true jika size matrix m1 sama dengan size matrix m2.

r. public int countElmt()

menghasilkan jumlah elemen pada sebuah matrix

s. public static Matrix fileToMatrix(String fileName)

Membaca sebuah file lalu memasukannya ke dalam matrix. Prosesnya yaitu dengan mencari baris terbesar dan kolom dari matrix dalam file. Lalu diubah ke double dan mengisikannya kedalam matrix m1.

Pada kasus tertentu seperti di bicubic dan interpolasi, ia akanmengisi sisanya dengan 0 pada baris yang kolomnya tidak sama banyak.

t. public static void saveMatrix(Matrix m1)

menyimpan hasil operasi yang berupa matriks ke dalam sebuah file. Akan meminta path dari file yang dijadikan tempat menyimpan matriks serta juga meminta konfirmasi dari user dalam menyimpan sampai benar.

u. public static void saveString(String s)

menyimpan hasil operasi yang berupa string ke dalam sebuah file. Akan meminta path dari file yang dijadikan tempat menyimpan matriks serta juga memiinta knfirmasi dari user dalam menyimpan sampai benar

v. public Matrix kofaktor(Matrix m1)

menghasilkan kofaktor dari sebuah matriks m1.

w. public Matrix adjoint(Matrix m1)

menghasilkan adjoint dari sebuah matriks m1.

x. public static Matrix mergeMatrix(Matrix m1, Matrix m2)

I.S. row kedua matriks dipastikan sama

menggabungkan dua buah matriks dengan membuat wadah berupa matriks dengan elemen kolom yaitu jumlah kolom matriks m1 dan matriks m2

y. public void addRow(Matrix m1, int row1, int row2, double l)

mengjumlahkan elemen baris dari matriks m1 yaitu baris row1 dan row2 yagn dikali dengan suatu nilai l.

z. public Matrix buatIdentitas(int k)

akan dibuat matriks identitas dengan ukuran k

aa. public void splitMatrix(Matrix m1, Matrix m2, int col)

memecah matriks augmented yang dimasukkan manjadi m1 yaitu matriks dari kolom 0 sampai kolom col-1 dan m2 yaitu matriks dari kolom col sampai kolom terkahir.

bb. public void inputSquareMatrix()

meminta user untuk memasukkan input berupa ukuran matriks persegi yang akan dibuat yang kemudian akan diisi oleh user sendiri.

cc. public static Matrix hilbert(int n)

membuat matriks hilbert sesuai aturan matriks hilbert

dd. public static void mainHilbert(String[] args)

menjalankan fungsi hilbert, meminta masukan kepada pengguna berapa besar matriks hilbert yang akan dibuat dan memasukannay dalam file.

B. Program

1. BicubicMenu.java

- Method :

a. private static double[] fungsi1(double x, double y)

Akan membuat fungsi bikubik f(x,y), keluarannya dalam bentuk array yang berisi koefisien tiap a_{ij} dimana i,j <= 3

b. private static double[] fungsiX(double x, double y)

Akan membuat fungsi bikubik turunan f(x,y) terhadap x, keluarannya dalam bentuk array yang berisi koefisien tiap a_{ij} dimana i,j <= 3

c. private static double[] fungsiY(double x, double y)

Akan membuat fungsi bikubik turunan f(x,y) terhadap y, keluarannya dalam bentuk array yang berisi koefisien tiap a_{ij} dimana i,j <= 3

d. private static double[] fungsiXY(double x, double y)

Akan membuat fungsi bikubik turunan f(x,y) terhadap xy, keluarannya dalam bentuk array yang berisi koefisien tiap aij dimana i,j <= 3

e. public static double[] splGaussBicubic(Matrix m1)

Melakukan proses gauss untuk mendapatkan nilai a_{ij}. Keluarannya berupa array berisi nilai $a_{ij.}$

f. public static void bikubik()

Melakukan proses input path file. Apabila path file benar maka program akan menyimpannya dalam matrix fromFile. Pada baris ke 5 matriks fromFile merupakan x dan y yang ditanyakan. x dan y yang ditanyakan itu disimpan dalam variabel a untuk x dan b untuk y.

Lalu program akan membuat matriks f sebesar 16x16. Matriks f akan di merge dengan matrix tambahan 16x1. Matrix tambahan ini merupakan convert dari matrix fromFile. Setelah di merge, akan dilakukan proses spl menggunakan splGaussBicubic yang menghasilkan array l_nilaiA berupa nilai setiap. Nantinya tiap elemen l_nilaiA akan dikaliakan dengan tiap elemen l_final yg merupakan hasil dari f(a,b).

2. DeterminanMenu.java

- Method :

a. public static void main(String[] args)

Menjalankan proses input dari user yang akan memilih menggunakan keyboard atau file. Bila memilih keyboard maka user harus memasukan matriks persegi. Jika pilihan dari file, maka user input path dari file tersebut. Setelah itu user dapat memilih akan menggunakan metode kofaktor atau invers. Jika sudah, maka user akan diminta untuk save atau tidak.

3. InterpolasiMenu.java

- Method :

a. public static void Interpolasi()

Fungsi ini mengeluarkan persamaan polinomial berdasarkan input derajat dan titik-titik dari keyboard serta mengeluarkan hasil f(x) yang ingin diketahui.

b. public static void InterpolasiFile(Matrix n)

Fungsi ini mengeluarkan persamaan polinomial berdasarkan input derajat dan titik-titik dari file serta mengeluarkan hasil f(x) dengan x dari row yang paling bawah pada file.

c. public static void Interpolasimenu()

Fungsi ini menjalankan aplikasi interpolasi.

4. InversMenu.java

- Method :

a. public static boolean fromFile()

Mengembalikan true jika pengguna ingin memasukkan input dari file, dan false jika sebaliknya

b. public static void menu()

Menampilkan pilihan-pilihan metode, yaitu:

1. Metode Eliminasi Gauss Jordan
2. Metode Kofaktor

dan menerima input pilihan metode, menampilkan pilihan input, yaitu

1. Keyboard, akan meminta path dari file
2. File, akan meminta ukuran matriks dan natriks itu sendiri

menerima pilihan input, mencetak hasil Matrix balikan berdasarkan metode yang dipilih user. Kemudian menerima input pilihan save ke dalam file (y/n)

5. MainMenu.java

- Method :

a. public static void main(String[] args)

Meminta input dari user nomer 1 sampai 7. Jika yang diinput diluar dari nomor tersebut, maka program akan meminta ulang masukannya. Setelah itu program akan menjalankan fungsi dari masing masing nomor. Jika user memilih nomor 7 di awal maka program akan selesai.

Setelah fungsi dari suatu nomor (1-6) telah dijalankan, program akan memunculkan 2 pilhan. Yaitu untuk kembali ka menu utama atau keluar. Jika

memilih untuk kembali ke menu utama, maka prgram akan menjalankan kembali fungsi main() sehingga user dapat memilih kembali persoalan nomor 1-7. Jika user memilih keluar, maka program akan selesai dijalankan.

6. RLBMenu.java

- Method :

a. public static void readKeyboard(Matrix input, Matrix x)

Membaca semua nilai-nilai x1i, x2i, ..., xni, nilai yi dari keyboard, dan menyimpannya ke Matrix input, serta membaca nilai-nilai xk yang akan ditaksir nilai fungsinya dari keyboard dan menyimpannya ke Matrix x.

b. public static void fileToMatrix2(Matrix input, Matrix x)

Membaca semua nilai-nilai x1i, x2i, ..., xni, nilai yi dari file dan menyimpannya ke Matrix input, serta membaca nilai-nilai xk yang akan ditaksir nilai fungsinya dari keyboard dan menyimpannya ke Matrix x.

c. public static Matrix toRLB(Matrix m1)

Membuat normal estimastion equation berdasarkan Matrix m1 dalam bentuk augmented matrix dan mengembalikan Matrix eselon tereduksi dari augmented matrix tersebut.

d. public static void outputRLB(Matrix m1, Matrix x)

I.S. : Matrix m1 terdefinisi sebagai augmented matrix yang memuat koefisien persamaan regresi dan Matrix x terdefinisi sebagai matrix yang memuat nilai-nilai xk yang akan ditaksir nilai fungsinya.

F.S. : Persamaan regresi dan hasil taksiran ditampilkan di layar dan atau tersimpan ke dalam sebuah file jika pengguna memilih untuk menyimpan hasil ke file.

e. public static void menu()

Menerima input pilihan metode, menampilkan pilihan input, yaitu

1. Keyboard, akan meminta path dari file
2. File, akan meminta ukuran matriks dan natriks itu sendiri

menerima pilihan input, mencetak hasil persamaan regresi dan hasil taksiran. Kemudian menerima input pilihan save ke dalam file (y/n)

7. SPLMenu.java

- Method :

  a. public static boolean checkMatrix(Matrix n)

  Fungsi ini mencari apakah ada baris yang anggotanya bernilai nol semua kecuali nilai b (kolom terakhir). Mengembalikan true jika ada dan false jika tidak.

  public static int zeroRow(Matrix n)

  Fungsi ini mengembalikan jumlah baris yang semua nilainya nol dari matriks augmented n.

  b. public static int getOther(double[] l)

  Fungsi ini mengembalikan banyaknya nilai bukan nol di antara kolom letak satu pertama dan kolom terakhir.

  c. public static void splGauss(Matrix m1)

  Prosedur ini menyelesaikan persamaan dalam bentuk matriks augmented m1 dengan metode eliminasi gauss.

  I.S. matriks m1 merupakan matriks augmented.

  F.S. mengeluarkan semua solusi nilai x dan mengeluarkan tidak ada penyelesaian jika tidak ada solusi.

  d. public static void splGauss(Matrix m1)

  Prosedur ini menyelesaikan persamaan dalam bentuk matriks augmented m1 dengan metode eliminasi gauss-jordan.

  I.S. matriks m1 merupakan matriks augmented.

  F.S. mengeluarkan semua solusi nilai x dan mengeluarkan tidak ada penyelesaian jika tidak ada solusi.

BAB IV

Eksperimen

1. **Temukan solusi SPL $Ax = b$, berikut:**

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

Output :

```
Sistem Persamaan Linear
1. Keyboard input
2. File input
Masukkan pilihan input: 2

Input nama file anda :
C:\Users\Dimas\Documents\GitHub\Tubes-Algeo-Apick\Test\Input\1a.txt
Pilih metode penyelesaian:
1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Metode matriks balikan
4. Kaidah Cramer

4
SPL memiliki banyak solusi atau tidak memiliki solusi. Coba metode lain.

Save hasil ke file (Y/N)?
y
Masukan nama file: C:\Users\Dimas\Documents\GitHub\Tubes-Algeo-Apick\Test\Output\1a_out.txt
Hasil sudah tersimpan di dalam file.
PS C:\Users\Dimas\Documents\GitHub\Tubes-Algeo-Apick>
```

```
Sistem Persamaan Linear
1. Keyboard input
2. File input
Masukkan pilihan input: 2

Input nama path file anda :
C:\Users\abdul\OneDrive\Documents\GitHub\Tubes-Algeo-Apick\Test\Input\1a.txt
Pilih metode penyelesaian:
1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Metode matriks balikan
4. Kaidah Cramer

1
Persamaan tidak memiliki penyelesaian.
Save hasil ke file (Y/N)?
Y
Masukan nama path file:
```

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

Output :

```
Sistem Persamaan Linear
1. Keyboard input
2. File input
Masukkan pilihan input: 2

Input nama path file anda :
C:\Users\abdul\OneDrive\Documents\GitHub\Tubes-Algeo-Apick\Test\Input\1b.txt
Pilih metode penyelesaian:
1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Metode matriks balikan
4. Kaidah Cramer

1
Solusi dari persamaan adalah:
x1 = 3.000000 + x5
x2 = 2.000000x5
x4 = -1.000000 + x5
```

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

c.

Output :

```
Sistem Persamaan Linear
1. Keyboard input
2. File input
Masukkan pilihan input: 2

Input nama path file anda :
C:\Users\abdul\OneDrive\Documents\GitHub\Tubes-Algeo-Apick\Test\Input\1c.txt
Pilih metode penyelesaian:
1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Metode matriks balikan
4. Kaidah Cramer

1
0.0000 1.0000 0.0000 0.0000 0.0000 1.0000 1.0000
0.0000 0.0000 0.0000 1.0000 0.0000 1.0000 -2.0000
0.0000 0.0000 0.0000 0.0000 1.0000 -1.0000 1.0000
Solusi dari persamaan adalah:
x2 = 1.000000 - x6
x4 = -2.000000 - x6
x5 = 1.000000 + x6
```

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

d.

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

Untuk n = 6

Output :

```
Sistem Persamaan Linear
1. Keyboard input
2. File input
Masukkan pilihan input: 2

Input nama path file anda :
C:\Users\Keanu\Documents\GitHub\Tubes-Algeo-Apick\Test\Input\1d.txt
Pilih metode penyelesaian:
1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Metode matriks balikan
4. Kaidah Cramer

1
1.0000 0.8572 0.7499 0.6665 0.5999 0.5453 0.0000
0.0000 1.0000 1.4898 1.7475 1.8703 1.9087 0.0000
0.0000 0.0000 1.0000 2.3300 3.2420 3.7793 0.0000
0.0000 0.0000 0.0000 1.0000 1.3216 1.4680 0.0000
0.0000 0.0000 0.0000 0.0000 1.0000 1.5976 0.0000
0.0000 0.0000 0.0000 0.0000 0.0000 1.0000 -327.1719
Solusi dari persamaan adalah:
x1 = 9.5965
x2 = -33.5491
x3 = 32.4251
x4 = -210.5076
x5 = 522.6862
x6 = -327.1719

Save hasil ke file (Y/N)?
y
Masukan nama path file: C:\Users\Keanu\Documents\GitHub\Tubes-Algeo-Apick\Test\Output\1d__out.txt
Hasil sudah tersimpan di dalam file.
1. Kembali ke Menu Utama
2. Keluar
Masukan pilihan anda:
```

Untuk n = 6

Output :

```
Sistem Persamaan Linear
1. Keyboard input
2. File input
Masukkan pilihan input: 2

Input nama path file anda :
C:\Users\Keanu\Documents\GitHub\Tubes-Algeo-Apick\Test\Input\1d2.txt
Pilih metode penyelesaian:
1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Metode matriks balikan
4. Kaidah Cramer

1
1.0000 0.9090 0.8330 0.7690 0.7140 0.6670 0.6250 0.5880 0.5560 0.5260 0.0000
0.0000 1.0000 1.6631 2.1577 2.4501 2.7313 2.7654 2.8556 3.0019 2.8676 0.0000
0.0000 0.0000 1.0000 3.0029 3.1838 5.6713 3.4676 5.5233 7.8020 3.9100 0.0000
0.0000 0.0000 0.0000 1.0000 3.9256 3.5053 7.7840 4.0488 6.2787 10.7406 0.0000
0.0000 0.0000 0.0000 0.0000 1.0000 0.8844 2.4717 1.7017 1.3026 3.6483 0.0000
0.0000 0.0000 0.0000 0.0000 0.0000 1.0000 0.1178 1.8076 0.5773 0.6865 0.0000
0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 1.0000 -1.6910 2.6308 0.0300 0.0000
0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 1.0000 -0.5948 0.5657 0.0000
0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 1.0000 -0.6307 0.0000
0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 1.0000 141.5339
Solusi dari persamaan adalah:
x1 = 20.1501
x2 = -167.6001
x3 = 340.9188
x4 = -212.2386
x5 = 175.6646
x6 = -66.3984
x7 = -284.6918
x8 = -26.9715
x9 = 89.2671
x10 = 141.5339

Save hasil ke file (Y/N)?
y
Masukan nama path file: C:\Users\Keanu\Documents\GitHub\Tubes-Algeo-Apick\Test\Output\1d2_out.txt
Hasil sudah tersimpan di dalam file.
1. Kembali ke Menu Utama
2. Keluar
Masukan pilihan anda:
```

2. SPL berbentuk matriks *augmented*

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

Output :

```
Sistem Persamaan Linear
1. Keyboard input
2. File input
Masukkan pilihan input: 2

Input nama path file anda :
C:\Users\abdul\OneDrive\Documents\GitHub\Tubes-Algeo-Apick\Test\Input\2a.txt
Pilih metode penyelesaian:
1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Metode matriks balikan
4. Kaidah Cramer

1
Solusi dari persamaan adalah:
x1 = -1.000000 + x4
x2 = 2.000000x3
```

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

Output :

```
Sistem Persamaan Linear
1. Keyboard input
2. File input
Masukkan pilihan input: 2

Input nama path file anda :
C:\Users\abdul\OneDrive\Documents\GitHub\Tubes-Algeo-Apick\Test\Input\2b.txt
Pilih metode penyelesaian:
1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Metode matriks balikan
4. Kaidah Cramer

1
Solusi dari persamaan adalah:
x1 = 0.000000
x2 = 2.000000
x3 = 1.000000
x4 = 1.000000
```

3. SPL berbentuk

a.

$$8x_1 + x_2 + 3x_3 + 2x_4 = 0$$

$$2x_1 + 9x_2 - x_3 - 2x_4 = 1$$

$$x_1 + 3x_2 + 2x_3 - x_4 = 2$$

$$x_1 + 6x_3 + 4x_4 = 3$$

Output :

```
Sistem Persamaan Linear
1. Keyboard input
2. File input
Masukkan pilihan input: 2

Input nama file anda :
C:\Users\Dimas\Documents\GitHub\Tubes-Algeo-Apick\Test\Input\3a.txt
Pilih metode penyelesaian:
1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Metode matriks balikan
4. Kaidah Cramer

4
x1 = -0.2243
x2 = 0.1824
x3 = 0.7095
x4 = -0.2581

Save hasil ke file (Y/N)?
n
Hasil tidak tersimpan
PS C:\Users\Dimas\Documents\GitHub\Tubes-Algeo-Apick>
```

b.

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

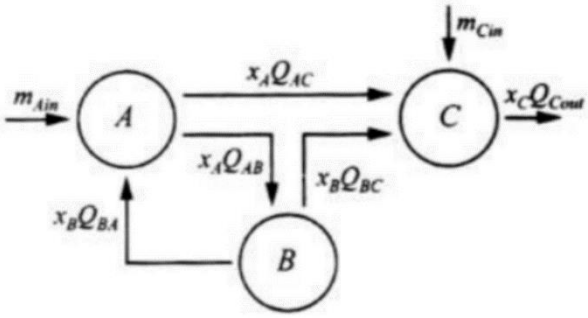
Output :

```
Sistem Persamaan Linear
1. Keyboard input
2. File input
Masukkan pilihan input: 2

Input nama path file anda :
C:\Users\abdul\OneDrive\Documents\GitHub\Tubes-Algeo-Apick\Test\Input\3b.txt
Pilih metode penyelesaian:
1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Metode matriks balikan
4. Kaidah Cramer

1
Persamaan tidak memiliki penyelesaian.
```

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:

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

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

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

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.

Output :

```
Sistem Persamaan Linear
1. Keyboard input
2. File input
Masukkan pilihan input: 2

Input nama file anda :
C:\Users\Dimas\Documents\GitHub\Tubes-Algeo-Apick\Test\Input\4.txt
Pilih metode penyelesaian:
1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Metode matriks balikan
4. Kaidah Cramer

4
x1 = 14.4444
x2 = 7.2222
x3 = 10.0000

Save hasil ke file (Y/N)?
n
Hasil tidak tersimpan
PS C:\Users\Dimas\Documents\GitHub\Tubes-Algeo-Apick>
```

5. Studi Kasus Interpolasi

a.

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:

$x = 0.2$ $f(x) = 0.0330$

$x = 0.55$ $f(x) = 0.1711$

$x = 0.85$ $f(x) = 0.3372$

$x = 1.28$ $f(x) = 0.6775$

Output :

```
Interpolasi Polinomial
1. Keyboard input
2. File input
Masukkan pilihan input: 2

Input path file anda :
C:\Users\abdul\OneDrive\Documents\GitHub\Tubes-Algeo-Apick\Test\Input\5a.txt
0.0000 0.0000 0.0001 0.0010 0.0100 0.1000 1.0000 0.0030
0.0007 0.0024 0.0081 0.0270 0.0900 0.3000 1.0000 0.0670
0.0156 0.0313 0.0625 0.1250 0.2500 0.5000 1.0000 0.1480
0.1176 0.1681 0.2401 0.3430 0.4900 0.7000 1.0000 0.2480
0.5314 0.5905 0.6561 0.7290 0.8100 0.9000 1.0000 0.3700
1.7716 1.6105 1.4641 1.3310 1.2100 1.1000 1.0000 0.5180
4.8268 3.7129 2.8561 2.1970 1.6900 1.3000 1.0000 0.6970
Persamaan hasi interpolasi:
y =  + 0.0260x^4 + 0.1974x^2 + 0.2400x^1 - 0.0230
Hasil dari f(0.200000) adalah 0.0330
Hasil dari f(0.550000) adalah 0.1711
Hasil dari f(0.850000) adalah 0.3372
Hasil dari f(1.280000) adalah 0.6775
```

b.

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 -> 33880.9320
b. 10/08/2022 -> 36978.2298
c. 05/09/2022 -> 345202.5540
d. Masukan user lainnya berupa tanggal (desimal) yang sudah diolah dengan asumsi prediksi selalu dilakukan untuk tahun 2022.

   08/07/2004 (7.266667) -> 39577.7943

Output :

```
Interpolasi Polinomial
1. Keyboard input
2. File input
Masukkan pilihan input: 2

Input path file anda :
C:\Users\abdul\OneDrive\Documents\GitHub\Tubes-Algeo-Apick\Test\Input\5b.txt
Persamaan hasi interpolasi:
y = 59952.4363x^9 + 3779703.1215x^8 - 104070008.9527x^7 + 1634525570.8092x^6 - 16016642894.8093x^5 + 100267994693.7373x^4 - 391617298893.4207x^3 + 8
72468595159.6174x^2 - 848868108974.3040x^1 + 236673.1058
Hasil dari f(7.225806) adalah 33880.9320
Hasil dari f(8.322581) adalah 36978.2298
Hasil dari f(9.166667) adalah 345202.5540
Hasil dari f(7.266667) adalah 39577.7943
```

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

Nilai x yang dicari: 1, 1.7

Output :

```
Interpolasi Polinomial
1. Keyboard input
2. File input
Masukkan pilihan input: 2

Input path file anda :
C:\Users\abdul\OneDrive\Documents\GitHub\Tubes-Algeo-Apick\Test\Input\5c.txt
0.0000 0.0000 0.0000 0.0000 0.0000 1.0000 0.0000
0.0102 0.0256 0.0640 0.1600 0.4000 1.0000 0.4189
0.3277 0.4096 0.5120 0.6400 0.8000 1.0000 0.5072
2.4883 2.0736 1.7280 1.4400 1.2000 1.0000 0.5609
10.4858 6.5536 4.0960 2.5600 1.6000 1.0000 0.5837
32.0000 16.0000 8.0000 4.0000 2.0000 1.0000 0.5767
Persamaan hasi interpolasi:
y = 0.2363x^5 - 1.4213x^4 + 3.2371x^3 - 3.5527x^2 + 2.0353x^1
Hasil dari f(1.000000) adalah 0.5347
Hasil dari f(1.700000) adalah 0.5806
```

6. Studi Kasus Regresi Linear Berganda

Diberikan sekumpulan data sesuai pada tabel berikut ini.

Table 12.1: Data for Example 12.1

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

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

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

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

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

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

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

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

Output :

```
REGRESI LINEAR BERGANDA
1. Keyboard input
2. File input
Masukkan pilihan input: 2

Masukan nama file:C:\Users\Dimas\Documents\GitHub\Tubes-Algeo-Apick\Test\Input\6.txt
f(x) = -3.5078 + -0.0026x1 + 0.0008x2 + 0.1542x3
f( 50.00 76.00 29.30 ) = 0.94
Save hasil ke file (Y/N)?
y
Masukan nama file: C:\Users\Dimas\Documents\GitHub\Tubes-Algeo-Apick\Test\Output\6_out.txt
Hasil sudah tersimpan di dalam file.
PS C:\Users\Dimas\Documents\GitHub\Tubes-Algeo-Apick>
```

$f(50, 76, 29.30) = 0.94$

7. 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:

$f(0, 0)$ = 21

Output :

```
INTERPOLASI BICUBIC SPLINE
Masukan nama path file: C:\Users\Keanu\Documents\GitHub\Tubes-Algeo-Apick\Test\Input\7a.txt

f(0.000000,0.000000) = 21.000000

Save hasil ke file (Y/N)?
y
Masukan nama path file: C:\Users\Keanu\Documents\GitHub\Tubes-Algeo-Apick\Test\Output\7a_out.txt
Hasil sudah tersimpan di dalam file.

1. Kembali ke Menu Utama
2. Keluar
Masukan pilihan anda:
```

$f(0.5, 0.5)$ = 87.796875

Output:

```
INTERPOLASI BICUBIC SPLINE
Masukan nama path file: C:\Users\Keanu\Documents\GitHub\Tubes-Algeo-Apick\Test\Input\7b.txt

f(0.500000,0.500000) = 87.796875

Save hasil ke file (Y/N)?
Y
Masukan nama path file: C:\Users\Keanu\Documents\GitHub\Tubes-Algeo-Apick\Test\Output\7b_out.txt
Hasil sudah tersimpan di dalam file.

1. Kembali ke Menu Utama
2. Keluar
Masukan pilihan anda:
```

$f(0.25, 0.75)$ = 117.732178

Output:

```
INTERPOLASI BICUBIC SPLINE
Masukan nama path file: C:\Users\Keanu\Documents\GitHub\Tubes-Algeo-Apick\Test\Input\7c.txt

f(0.250000,0.750000) = 117.732178

Save hasil ke file (Y/N)?
Y
Masukan nama path file: C:\Users\Keanu\Documents\GitHub\Tubes-Algeo-Apick\Test\Output\7c_out.txt
Hasil sudah tersimpan di dalam file.

1. Kembali ke Menu Utama
2. Keluar
Masukan pilihan anda:
```

$f(0.1, 0.9)$ = 128.575187

Output :

```
INTERPOLASI BICUBIC SPLINE
Masukan nama path file: C:\Users\Keanu\Documents\GitHub\Tubes-Algeo-Apick\Test\Input\7d.txt

f(0.100000,0.900000) = 128.575187

Save hasil ke file (Y/N)?
Y
Masukan nama path file: C:\Users\Keanu\Documents\GitHub\Tubes-Algeo-Apick\Test\Output\7d_out.txt
Hasil sudah tersimpan di dalam file.

1. Kembali ke Menu Utama
2. Keluar
Masukan pilihan anda:
```

BAB V

Kesimpulan

Operasi matriks, seperti menghitung determinan, penentuan invers matriks dengan metode eliminasi Gauss, dan eliminasi Gauss Jordan dapat digunakan untuk menyelesaikan masalah - masalah yang melibatkan matriks atau persamaan. Salah satu contohnya adalah mencari nilai variabel – variabel dalam Sistem Persamaan Linear. Sistem Persamaan Linear sangat banyak dipakai dalam kehidupan sehari – hari, hal ini juga menuntut penyelesaian Sistem Persamaan Linear yang cepat dan akurat. Hal tersebut merupakan salah satu tujuan program ini dibuat. Dari satu sistem persaman linear terapat 3 jenis penyelesaian, yaitu banyak solusi, tidak ada solusi, atau satu solusi unik. Terdapat 4 metode penyelesaian SPL yang dibahas dalam laporan ini yang sesuai dengan spesifikasi, yaitu metode Gauss, metode Gauss-Jordan, metode Invers, dan kaidah Cramer. Pada beberapa kasus yang dibahas pada BAB IV, terdapat beberapa SPL yang tidak dapat diselesaikan dengan metode Invers ataupun kaidah Cramer karena kedua metode ini hanya dapat menyelesaikan permasalahan matriks segiempat dengan determinan bukan 0, maka dari itu kami menambahkan kalimat “coba metode lain” karena mungkin dengan metode Gauss serta Gauss-Jordan SPL tersebut memiliki penyelesaian yang berupa persamaa parametrik. SPL kemudian dapat dimanfaatkan untuk melakukan Interpolasi Polinom, Interpolasi Bikubik, dan Regresi Linear Berganda. Interpolasi Polinom dapat digunakan untuk membuat sebuah persamaan polinom dari n buah titik yang diketahui dan memprediksi nilai-nilai yang tidak diketahui. Dalam BAB IV, interpolasi polinom digunakan untuk memprediksi jumlah kasus Covid-19 pada tanggal tertentu dan juga untuk menyederhakan sebuah fungsi Matematika. Interpolasi Bikubik digunakan untuk memprediksi nilai-nilai pada titik tertentu dengan mempertimbangkan 16 nilai yang berada di sekitar titik tersebut. Regresi Linear Berganda digunakan untuk memprediksi nilai sebuah variabel yang dipengaruhi oleh beberapa peubah lainnya. Dalam BAB IV, RLB digunakan untuk memprediksi Nitrous Oxide berdasarkan Humidity, Pressure, dan Temperature.

Saran

Saran dari penulis untuk selanjutnya adalah format laporan dalam spesifikasi tubes bisa diperjelas lagi jika perlu diberi format dalam bentuk dokumen.

Refleksi

Secara garis besar, tim penulis dapat mengerjakan tugas besar ini dengan baik dalam kurun waktu yang telah ditentukan, tim penulis juga sempat mengerjakan bonus. Selain

memperdalam materi kuliah dan implementasinya dalam program, tim penulis juga mendapatkan kesempatan untuk memelajari beberapa metode baru seperti materi baru, seperti interpolasi bikubik dan regresi linear berganda. Tim penulis berhasil membuat program ini berjalan dengan baik dengan memperhatikan masukan dari pengguna yang tidak sesuai.

Referensi

Munir, Rinaldi. 2023 "Sistem persamaan linier (Bagian 1: Metode eliminasi Gauss)(Update 2023)" Diakses 29 September 2023, dari

https://informatika.stei.itb.ac.id/~rinaldi.munir/AljabarGeometri/2023-2024/Algeo-03-Sistem-Persamaan-Linier-2023.pdf

Munir, Rinaldi. 2023 "Sistem persamaan linier (Bagian 2: Tiga kemungkinan solusi sistem persamaan linier (Update 2023)" Diakses 29 September 2023, dari

https://informatika.stei.itb.ac.id/~rinaldi.munir/AljabarGeometri/2023-2024/Algeo-04-Tiga-Kemungkinan-Solusi-SPL-2023.pdf