

LAPORAN TUGAS BESAR 1
IF2123 ALJABAR LINIER DAN GEOMETRI
Sistem Persamaan Linier, Determinan, dan Aplikasinya



Disusun oleh:

Grup RGB (04)

Nama	NIM
Refki Alfarizi	13523002
Adhimas Aryo Bimo	13523052
Guntara Hambali	13523114

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

2024

Daftar Isi

Bab 1 : Deskripsi Masalah.....	1
1. Abstraksi.....	1
1.1. Interpolasi Polinomial.....	1
1.2. Regresi Berganda.....	3
1.2.1. Regresi Linier Berganda.....	3
1.2.2. Regresi Kuadratik Berganda.....	3
1.3. Bicubic Spline Interpolation.....	4
Bab 2 : Teori Singkat.....	7
2.1. Metode Baris Elementer.....	7
2.2. Metode Eliminasi Gauss.....	7
2.3. Metode Eliminasi Gauss-Jordan.....	7
2.4. Determinan.....	8
2.5. Matriks Balikan.....	9
2.6. Matriks Kofaktor.....	9
2.7. Adjoint Matrix.....	10
2.8. Kaidah Cramer.....	11
2.9. Interpolasi Polinom.....	11
2.10. Interpolasi Bicubic-Spline.....	12
2.11. Regresi Linear Berganda.....	12
2.12. Regresi Kuadratik Berganda.....	13
Bab 3 : Implementasi Pustaka dan Program dalam Java.....	14
3.1. Folder types.....	14
3.1.1. Matrix.....	14
3.1.2. PolynomialResult.....	17
3.1.3. UniqueSolution.....	18
3.1.4. ParametricSolution.....	19
3.1.5. NoSolution.....	20
3.1.6. LinearSystemSolution.....	20
3.2. Folder solvers.....	21
3.2.1. LinearSystemSolvers.....	21
3.3. Folder mathmodels.....	22
3.3.1. Interpolation.....	22
3.3.2. Regression.....	23
3.4. Folder imageresizing.....	24
3.4.1. Image Resizing.....	24
3.5. Folder cli (Command Line Interface).....	25
Folder ini merupakan program utama yang akan mengintegrasikan seluruh fitur pada aplikasi.	
Folder ini terdiri dari beberapa file menu dan submenu.....	25
3.6. Folder gui (Graphical User Interface).....	25
Folder ini merupakan pengembangan dari penerapan program utama. Pada folder ini, berisi file-file fxml untuk membuat layout dan elemen UI utama serta styling-nya. Selain itu, folder ini juga memuat file-file yang mengatur logika dari UI ketika diinteraksikan (diklik, diketik, dsb.)..	25

Bab 4 : Eksperimen.....	26
4.1. Studi dan Tes Kasus.....	26
Bab 5 : Kesimpulan, Saran, dan Refleksif.....	36
5.1. Kesimpulan.....	36
5.2. Saran.....	36
5.3. Komentar.....	36
5.4. Refleksi.....	36
LAMPIRAN.....	37

Bab 1 : Deskripsi Masalah

1. Abstraksi

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).

$$\left[\begin{array}{cccc} 0 & 2 & 1 & -1 \\ 0 & 0 & 3 & 1 \\ 0 & 0 & 0 & 0 \end{array} \right] . \quad \left[\begin{array}{cccc} 0 & 1 & 0 & -\frac{2}{3} \\ 0 & 0 & 1 & \frac{1}{3} \\ 0 & 0 & 0 & 0 \end{array} \right].$$

Gambar 1. Eliminasi Gauss dilakukan dengan matriks eselon baris dan eliminasi Gauss-Jordan dengan matriks eselon baris tereduksi.

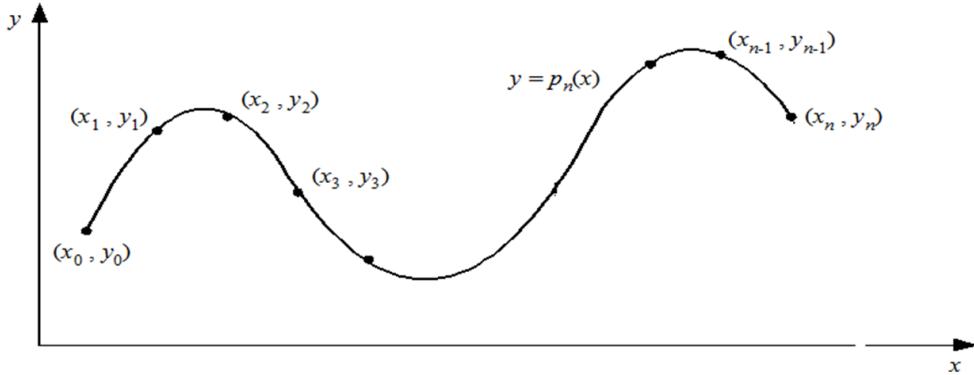
Di dalam Tugas Besar 1 ini, Anda diminta membuat satu atau lebih *library* aljabar linier dalam Bahasa Java. Library tersebut berisi fungsi-fungsi seperti eliminasi Gauss, eliminasi Gauss-Jordan, menentukan balikan matriks, menghitung determinan, kaidah Cramer (kaidah Cramer khusus untuk SPL dengan n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

Beberapa tulisan cara membuat library di Java:

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

1.1. 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 $i = 0, 1, 2, \dots, n$.



Gambar 1.1. Ilustrasi beberapa titik yang diinterpolasi secara polinomial.

Setelah polinom interpolasi $p_n(x)$ ditemukan, $p_n(x)$ dapat digunakan untuk menghitung perkiraan nilai y di sembarang titik di dalam selang $[x_0, x_n]$.

Polinom interpolasi derajat n yang menginterpolasi titik-titik $(x_0, y_0), (x_1, y_1), \dots, (x_n, y_n)$, adalah berbentuk $p_n(x) = a_0 + a_1x + a_2x^2 + \dots + a_nx^n$. Jika hanya ada dua titik, (x_0, y_0) dan (x_1, y_1) , maka polinom yang menginterpolasi kedua titik tersebut adalah $p_1(x) = a_0 + a_1x$ yaitu berupa persamaan garis lurus. Jika tersedia tiga titik, $(x_0, y_0), (x_1, y_1)$, dan (x_2, y_2) , maka polinom yang menginterpolasi ketiga titik tersebut adalah $p_2(x) = a_0 + a_1x + a_2x^2$ atau persamaan kuadrat dan kurvanya berupa parabola. Jika tersedia empat titik, $(x_0, y_0), (x_1, y_1), (x_2, y_2)$, dan (x_3, y_3) , polinom yang menginterpolasi keempat titik tersebut adalah $p_3(x) = a_0 + a_1x + a_2x^2 + a_3x^3$, demikian seterusnya. Dengan cara yang sama kita dapat membuat polinom interpolasi berderajat n untuk n yang lebih tinggi asalkan tersedia $(n+1)$ buah titik data. Dengan menyulihkan (x_i, y_i) ke dalam persamaan polinom $p_n(x) = a_0 + a_1x + a_2x^2 + \dots + a_nx^n$ untuk $i = 0, 1, 2, \dots, n$, akan diperoleh n buah sistem persamaan lanjar dalam $a_0, a_1, a_2, \dots, a_n$,

$$\begin{aligned} a_0 + a_1x_0 + a_2x_0^2 + \dots + a_nx_0^n &= y_0 \\ a_0 + a_1x_1 + a_2x_1^2 + \dots + a_nx_1^n &= y_1 \\ &\dots &&\dots \\ a_0 + a_1x_n + a_2x_n^2 + \dots + a_nx_n^n &= y_n \end{aligned}$$

Solusi sistem persamaan lanjar ini, yaitu nilai a_0, a_1, \dots, 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

$$\begin{aligned} a_0 + 8.0a_1 + 64.00a_2 &= 2.0794 \\ a_0 + 9.0a_1 + 81.00a_2 &= 2.1972 \\ a_0 + 9.5a_1 + 90.25a_2 &= 2.2513 \end{aligned}$$

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

1.2. 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.

1.2.1. 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:

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

1.2.2. 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:

1. Variabel Linier: Variabel dengan derajat satu seperti X, Y, dan Z
2. Variabel Kuadrat: Variabel dengan derajat dua seperti X^2
3. 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.

1.3. 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.

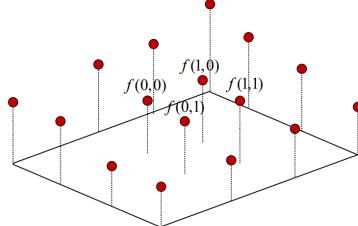
Dalam pemrosesan menggunakan interpolasi *bicubic spline* digunakan 16 buah titik, 4 titik referensi utama di bagian pusat, dan 12 titik di sekitarnya sebagai aproksimasi turunan dari keempat titik referensi untuk membagun permukaan bikubik. Bentuk pemodelannya adalah sebagai berikut.

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 1.3. Pemodelan interpolasi *bicubic spline*.

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

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

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

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

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

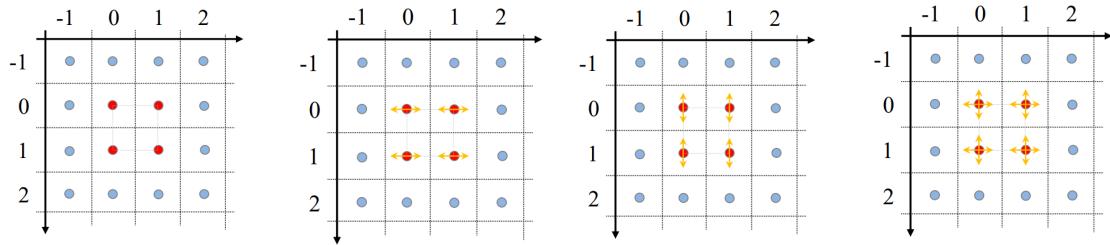
Dengan menggunakan nilai fungsi dan turunan berarah tersebut, dapat terbentuk sebuah matriks solusi X yang membentuk persamaan penyelesaian sebagai berikut.

$$\begin{matrix} y = Xa \\ \left[\begin{array}{c} f(0,0) \\ f(1,0) \\ f(0,1) \\ f(1,1) \\ f_x(0,0) \\ f_x(1,0) \\ f_x(0,1) \\ f_x(1,1) \\ f_y(0,0) \\ f_y(1,0) \\ f_y(0,1) \\ f_y(1,1) \\ f_{xy}(0,0) \\ f_{xy}(1,0) \\ f_{xy}(0,1) \\ f_{xy}(1,1) \end{array} \right] = \left[\begin{array}{cccccccccccccccccccccccc} 1 & 0 \\ 1 & 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 \\ 0 & 1 & 0 \\ 0 & 1 & 2 & 3 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 2 & 3 & 0 & 1 & 2 & 3 & 0 & 1 & 2 & 3 & 0 & 1 & 2 & 3 & 0 & 1 & 2 & 3 & 0 & 1 & 2 & 3 & 0 & 1 & 2 & 3 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 2 & 2 & 2 & 2 & 2 & 3 & 3 & 3 & 3 & 3 & 3 & 3 & 3 & 3 & 3 & 3 & 3 & 3 & 3 & 3 & 3 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 2 & 3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 2 & 3 & 0 & 2 & 4 & 6 & 0 & 3 & 6 & 9 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array} \right] \end{matrix}$$

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 4×4 . 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 4. Nilai fungsi yang akan di interpolasi pada titik merah, turunan berarah terhadap sumbu x , terhadap sumbu y , dan keduanya (kiri ke kanan).

Untuk studi kasus ini, buatlah matriks X menggunakan persamaan yang ada (tidak *hardcode*) serta carilah invers matriks X dengan *library* yang telah kalian buat dalam penyelesaian masalah. Berikut adalah [sebuah tautan](#) yang dapat dijadikan referensi.

Bab 2 : Teori Singkat

2.1. Metode Baris Elementer

Metode baris elementer adalah serangkaian operasi yang dilakukan pada baris-baris suatu matriks untuk mengubahnya menjadi bentuk yang lebih sederhana. Tiga operasi dasar yang dapat dilakukan meliputi:

- Penukaran baris: Menukar dua baris dalam suatu matriks.
- Perkalian baris dengan skalar non-nol: Mengalikan semua elemen suatu baris dengan suatu bilangan bukan nol.
- Penjumlahan baris: Menambahkan kelipatan suatu baris ke baris lainnya.

Operasi ini digunakan untuk memecahkan sistem persamaan linear, mencari invers matriks, dan menentukan rank suatu matriks.

Metode baris elementer merupakan dasar untuk melakukan beberapa metode yang akan dipakai dalam program kalkulator matriks. Metode ini akan menghasilkan matriks eselon baris atau matriks eselon tereduksi sesuai dengan solusi yang dihasilkan.

2.2. Metode Eliminasi Gauss

Eliminasi Gauss adalah suatu teknik untuk menyelesaikan sistem persamaan linear dengan mengubah matriks koefisien menjadi bentuk eselon baris (row echelon form) menggunakan operasi baris elementer. Tujuannya adalah menyederhanakan matriks sehingga dapat diperoleh solusi dari sistem tersebut melalui substitusi mundur.

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

Gambar 2.2 Eliminasi Gauss

2.3. Metode Eliminasi Gauss-Jordan

Metode eliminasi Gauss-Jordan adalah varian dari eliminasi Gauss yang bertujuan mengubah matriks menjadi bentuk eselon baris tereduksi (reduced row echelon form) atau matriks identitas. Proses ini memungkinkan kita menyelesaikan sistem persamaan linear dengan menemukan solusi yang unik (jika ada) atau menyatakan apakah sistem tersebut tak konsisten atau memiliki solusi tak terbatas.

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

Gambar 2.3 Eliminasi Gauss - Jordan

2.4. Determinan

Determinan adalah suatu nilai yang dapat dihitung dari suatu matriks persegi dan memberikan informasi penting mengenai sifat matriks tersebut, seperti apakah matriks tersebut dapat di-invers atau tidak.

$$\det(A) = \begin{vmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{vmatrix}$$

Gambar 2.4.1 Determinan

Determinan juga digunakan dalam Metode Reduksi Baris, Ekspansi Kofaktor, dan Metode Kaidah Cramer untuk menyelesaikan sistem persamaan linear.

$$\left[\begin{array}{cccc} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{array} \right] \underset{\text{OBE}}{\sim} \left[\begin{array}{cccc} a'_{11} & a'_{12} & \dots & a'_{1n} \\ 0 & a'_{22} & \dots & a'_{2n} \\ \vdots & \vdots & \ddots & a'_{3n} \\ 0 & 0 & 0 & a'_{nn} \end{array} \right]$$

Gambar 2.4.2 Determinan dengan Metode Reduksi Baris

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

Gambar 2.4.3 Determinan dengan Metode Kaidah Cramer

$$\det(A) = \sum_{j=1}^n a_{ij} C_{ij} = a_{i1} C_{i1} + a_{i2} C_{i2} + \dots + a_{in} C_{in}$$

Gambar 2.4.4. Determinan dengan Metode Ekspansi Kofaktor di Baris-i

2.5. Matriks Balikan

Matriks balikan adalah matriks yang, jika dikalikan dengan matriks asal, menghasilkan matriks identitas. Hanya matriks yang tidak singular (determinannya tidak nol) yang memiliki matriks invers. Matrik Balikan dapat dicari menggunakan dua metode, yakni menggunakan Adjoint dan Gauss-Jordan

Metode Adjoint dapat digunakan dengan mengaplikasikan rumus berikut:

$$M^{-1} = \frac{1}{|M|} adj(M)$$

Gambar 2.5.1 Matriks Balikan dengan Metode Adjoint

Metode Gauss-Jordan dapat digunakan dengan mengaplikasikan rumus berikut:

$$(A|I) \sim GJ \sim (I|A^{-1})$$

Gambar 2.5.2 Matriks Balikan dengan Metode Gauss-Jordan

2.6. Matriks Kofaktor

Kofaktor dari suatu elemen dalam matriks adalah determinan minor elemen tersebut, dengan tanda tertentu yang bergantung pada posisinya dalam matriks. Kofaktor digunakan dalam perhitungan determinan matriks yang lebih besar dan dalam menemukan invers matriks. Matriks kofaktor diisi oleh kofaktor itu sendiri menggunakan rumus berikut:

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

Gambar 2.6.1 Rumus Matriks Kofaktor

$$\mathbf{C} = \begin{bmatrix} +\begin{vmatrix} b_2 & b_3 \\ c_2 & c_3 \end{vmatrix} & -\begin{vmatrix} b_1 & b_3 \\ c_1 & c_3 \end{vmatrix} & +\begin{vmatrix} b_1 & b_2 \\ c_1 & c_2 \end{vmatrix} \\ -\begin{vmatrix} a_2 & a_3 \\ c_2 & c_3 \end{vmatrix} & +\begin{vmatrix} a_1 & a_3 \\ c_1 & c_3 \end{vmatrix} & -\begin{vmatrix} a_1 & a_2 \\ c_1 & c_2 \end{vmatrix} \\ +\begin{vmatrix} a_2 & a_3 \\ b_2 & b_3 \end{vmatrix} & -\begin{vmatrix} a_1 & a_3 \\ b_1 & b_3 \end{vmatrix} & +\begin{vmatrix} a_1 & a_2 \\ b_1 & b_2 \end{vmatrix} \end{bmatrix},$$

Gambar 2.6.2 Matriks Kofaktor

M_{ij} merupakan elemen minor pada matriks. Misalkan terdapat matriks A dengan ukuran 3×3 , maka akan didapatkan M_{12} seperti berikut:

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \quad M_{12} = \begin{bmatrix} a_{21} & a_{23} \\ a_{31} & a_{33} \end{bmatrix}$$

Gambar 2.6.3 Matriks Minor

2.7. Adjoint Matrix

Adjoin dari suatu matriks adalah transpose dari matriks kofaktornya. Matriks adjoint digunakan dalam perhitungan invers matriks, terutama untuk matriks yang lebih besar.

$$\text{adj}(A) = \mathbf{C}^T = \begin{bmatrix} +\begin{vmatrix} b_2 & b_3 \\ c_2 & c_3 \end{vmatrix} & -\begin{vmatrix} a_2 & a_3 \\ c_2 & c_3 \end{vmatrix} & +\begin{vmatrix} a_2 & a_3 \\ b_2 & b_3 \end{vmatrix} \\ -\begin{vmatrix} b_1 & b_3 \\ c_1 & c_3 \end{vmatrix} & +\begin{vmatrix} a_1 & a_3 \\ c_1 & c_3 \end{vmatrix} & -\begin{vmatrix} a_1 & a_3 \\ b_1 & b_3 \end{vmatrix} \\ +\begin{vmatrix} b_1 & b_2 \\ c_1 & c_2 \end{vmatrix} & -\begin{vmatrix} a_1 & a_2 \\ c_1 & c_2 \end{vmatrix} & +\begin{vmatrix} a_1 & a_2 \\ b_1 & b_2 \end{vmatrix} \end{bmatrix}.$$

Gambar 2.7 Adjoint Matrix

2.8. Kaidah Cramer

Kaidah Cramer adalah metode untuk menyelesaikan sistem persamaan linear dengan menggunakan determinan. Kaidah ini berlaku untuk sistem dengan jumlah persamaan dan variabel yang sama, dan memiliki solusi unik. Solusi diperoleh dengan membagi determinan dari matriks yang dihasilkan dari penggantian kolom dengan determinan matriks koefisien

Misalkan terdapat persamaan seperti berikut:

$$a_1 + b_1 = c_1 \quad \text{persamaan 1}$$

$$a_2 + b_2 = c_2 \quad \text{persamaan 2}$$

Didapatkan matriks $Ax = B$ seperti berikut:

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

Maka didapatkan determinan sebagai berikut:

$$D = |A| = \begin{vmatrix} a_1 & b_1 \\ a_2 & b_2 \end{vmatrix} = a_1b_2 - a_2b_1$$

$$D_x = \begin{vmatrix} c_1 & b_1 \\ c_2 & b_2 \end{vmatrix} = c_1b_2 - c_2b_1$$
$$D_y = \begin{vmatrix} a_1 & c_1 \\ a_2 & c_2 \end{vmatrix} = a_1c_2 - a_2c_1$$

Sehingga bisa didapatkan nilai-nilai tiap variabel dengan:

$$x = \frac{D_x}{D} \quad \text{dan} \quad y = \frac{D_y}{D}$$

2.9. Interpolasi Polinom

Interpolasi polinom adalah metode untuk menemukan suatu polinom yang melewati sekumpulan titik data. Polinom tersebut dapat digunakan untuk memperkirakan nilai di antara titik-titik data yang diketahui. Untuk menggunakan metode interpolasi polinom dibutuhkan data seminimalnya $n + 1$ titik untuk dapat membuat polinom interpolasi $p_n(x)$, setelah polinom ditemukan dapat ditaksir nilai y di sembarang titik didalam selang $[x_0, x_n]$.

$$\begin{bmatrix} x_0^n & x_0^{n-1} & x_0^{n-2} & \dots & x_0 & 1 \\ x_1^n & x_1^{n-1} & x_1^{n-2} & \dots & x_1 & 1 \\ \vdots & \vdots & \vdots & & \vdots & \vdots \\ x_n^n & x_n^{n-1} & x_n^{n-2} & \dots & x_n & 1 \end{bmatrix} \begin{bmatrix} a_n \\ a_{n-1} \\ \vdots \\ a_0 \end{bmatrix} = \begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_n \end{bmatrix}.$$

Gambar 2.9 Matriks Interpolasi Polinomial

2.10. Interpolasi Bicubic-Spline

Interpolasi bicubic spline adalah teknik interpolasi dua dimensi yang menghasilkan permukaan halus melalui suatu grid titik-titik data. Metode ini sering digunakan dalam pemrosesan citra dan grafika komputer untuk memperbesar gambar atau memperkirakan nilai di antara titik-titik data.

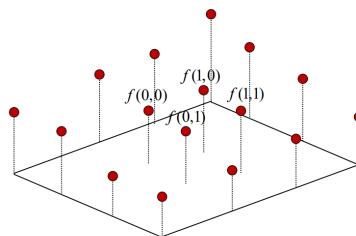
Dalam pemrosesan menggunakan interpolasi *bicubic spline* digunakan 16 buah titik, 4 titik referensi utama di bagian pusat, dan 12 titik di sekitarnya sebagai aproksimasi turunan dari keempat titik referensi untuk membagun permukaan bikubik. Bentuk pemodelannya adalah sebagai berikut.

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 2.10 Pemodelan Interpolasi Bicubic Spline

2.11. Regresi Linear Berganda

Regresi linear berganda adalah metode statistik yang digunakan untuk memodelkan hubungan antara variabel dependen dengan dua atau lebih variabel independen. Persamaan umum regresi linear berganda adalah:

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

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

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

Gambar 2.11 Normal Estimation untuk Regresi Linear Berganda

2.12. Regresi Kuadratik Berganda

Regresi kuadratik berganda adalah pengembangan dari regresi linear berganda, di mana modelnya memasukkan variabel-variabel kuadrat (pangkat dua) untuk menangkap hubungan yang melengkung antara variabel dependen dan variabel independen.

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

Gambar 2.12 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:

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

Bab 3 : Implementasi Pustaka dan Program dalam Java

3.1. Folder types

3.1.1. Matrix

Kelas Matrix merupakan kelas untuk membuat tipe data Matrix yang akan dipakai dalam pengoperasian Matrix.

a. Atribut

Atribut	Deskripsi
double[][] data	Menyimpan data dari matrix dua dimensi
int rows	Menyimpan data jumlah baris dalam matriks
int cols	Menyimpan data jumlah kolom dalam matriks

b. Konstruktor

Atribut	Deskripsi
Matrix(int rows, int cols)	Konstruktor untuk membuat matriks dengan ukuran baris sebanyak variabel rows dan ukuran kolom sebanyak variabel cols.

c. Selektor

Atribut	Deskripsi
public int getRowsCount()	Selektor untuk mengambil jumlah baris yang ada pada suatu Matrix
public int getColsCount()	Selektor untuk mengambil jumlah kolom yang ada pada suatu Matrix
public double[].getRow(int row)	Selektor untuk mengambil data di sebuah baris pada suatu Matrix
public double[][] getCol(int col)	Selektor untuk mengambil data di sebuah kolom pada suatu Matrix
public double[][] getAllData()	Selektor untuk mengambil semua data

	pada suatu Matrix
public double getData(int row, int col)	Selektor untuk mengambil data pada baris dan kolom tertentu pada suatu Matrix
public void setAllData(double[][] data)	Selektor untuk mengubah semua data pada suatu Matrix
public void setData(int row, int col, double val)	Selektor untuk mengubah data pada kolom dan baris tertentu pada suatu Matrix
public void setCol(int col, double[] colData)	Selektor untuk mengubah data baris pada suatu Matrix
public void setRow(int row, double[] rowData)	Selektor untuk mengubah data kolom pada suatu Matrix

d. Fungsi / Prosedur

Fungsi / Prosedur	Deskripsi
public Matrix add(Matrix other)	Menambahkan suatu matriks dengan matriks lainnya
public Matrix subtract(Matrix other)	Mengurangkan suatu matriks dengan matriks lainnya
public Matrix multiplyByScalar(double scalar)	Mengalikan suatu matriks dengan skalar.
public Matrix divideByScalar(double scalar)	Membagi suatu matriks dengan skalar.
public Matrix multiplyByMatrix(Matrix other)	Mengalikan suatu matriks dengan matriks lainnya
public Matrix multiplyRowByScalar(int row, double scalar)	Mengalikan suatu baris pada matriks dengan suatu skalar
public Matrix multiplyColByScalar(int col, double scalar)	Mengalikan suatu kolom pada matriks dengan suatu skalar
public Matrix transpose()	Mengembalikan transpose suatu matrix
public Matrix swapRow(int row1, int row2)	Menukar posisi dua baris pada matriks yang sama
public Matrix swapCol(int col1, int	Menukar posisi dua kolom pada matriks

col2)	yang sama
public double getDeterminantWithCofactor()	Mengembalikan determinan suatu matriks dengan metode minor-kofaktor
public double getDeterminantWithCofactor(Matrix matrix)	APA BEDANYA
public double getDeterminantWithRowReduction()	Mengembalikan determinan suatu matriks dengan mencari trace suatu matriks yang sudah dijadikan matriks eselon baris
public Matrix getRowEchelonForm ()	Mengembalikan matriks dalam bentuk eselon baris
public Matrix getReducedRowEchelonForm()	Mengembalikan matriks dalam bentuk eselon baris tereduksi
public Matrix getAdjoint()	Mengembalikan adjoint suatu matriks
public Matrix getInverseWithAdjoint()	Mengembalikan balikan suatu matriks menggunakan metode adjoint
public boolean isSquare()	Mengecek apakah suatu matriks persegi atau bukan
private static boolean isAllZero(double[] row)	Mengecek apakah suatu baris pada suatu matriks memiliki nilai semuanya nol
private void validateRowIndex(int row)	Mengecek apakah indeks baris yang dimasukkan valid. Jika tidak, <i>throw error</i>
private void validateColIndex(int col)	Mengecek apakah indeks kolom yang dimasukkan valid. Jika tidak, <i>throw error</i>
private void validateDimensions(Matrix other)	Mengecek apakah indeks kolom sekaligus baris yang dimasukkan valid. Jika tidak, <i>throw error</i>
public Matrix getCopy()	Mengembalikan salinan dari matriks
private double adjustPrecision(double value)	Mengembalikan nilai suatu skalar yang sudah disesuaikan presisinya
public Matrix getValidatedMatrixPrecision()	Mengembalikan matriks yang tiap komponennya sudah disesuaikan presisinya

public void printMatrix()	Mencetak matriks
public void inputMatrix()	Memasukkan nilai matriks dari <i>keyboard</i>
public void inputMatrixFromString(String input)	Memasukkan nilai matriks dari suatu string
public void inputMatrixFromFile(String fileName)	Memasukkan nilai matriks dari suatu file
public boolean equals(Object obj)	Mengecek apakah kedua matriks sama

3.1.2. PolynomialResult

a. Atribut

Atribut	Deskripsi
Matrix coefficients	Menyimpan koefisien hasil interpolasi polinomial

b. Konstruktor

Atribut	Deskripsi
public PolynomialResult(Matrix coefficients)	Konstruktor untuk membuat kelas PolynomialResult dengan masukan adalah matriks koefisien hasil interpolasi polinomial

c. Selektor

Atribut	Deskripsi
public Matrix getCoefficients()	Mendapatkan koefisien hasil interpolasi polinomial

d. Fungsi / Prosedur

Atribut	Deskripsi
public double evaluate(double x)	Mengembalikan hasil prediksi suatu x berdasarkan koefisien hasil interpolasi polinomial

3.1.3. UniqueSolution

a. Atribut

Atribut	Deskripsi
Matrix solution	Menyimpan solusi unik dari suatu sistem persamaan linear dalam bentuk matriks

b. Konstruktor

Atribut	Deskripsi
public UniqueSolution(Matrix solution)	Membentuk kelas UniqueSolution dengan data solusi unik dari suatu sistem persamaan linear

c. Selektor

Atribut	Deskripsi
public Matrix getSolution()	Mendapatkan solusi unik dari suatu sistem persamaan linear dalam bentuk matriks

d. Fungsi / Prosedur

Atribut	Deskripsi
public String toString()	Mengembalikan string hasil solusi unik yang siap untuk dicetak (<i>output</i>)

3.1.4. ParametricSolution

a. Atribut

Atribut	Deskripsi
String[][] parametricForm	Menyimpan solusi parametrik dari suatu sistem persamaan linear dalam bentuk matriks

b. Konstruktor

Atribut	Deskripsi
public ParametricSolution(String[][] parametricForm)	Membentuk kelas ParametricSolution dengan data solusi parametrik dari suatu sistem persamaan linear

c. Selektor

Atribut	Deskripsi
public String[][] getParametricForm()	Mendapatkan data solusi parametrik dari suatu sistem persamaan linear dalam bentuk matriks

d. Fungsi / Prosedur

Atribut	Deskripsi
public String toString()	Mengembalikan string hasil solusi parametrik yang siap untuk dicetak (<i>output</i>)

3.1.5. NoSolution

a. Atribut

Atribut	Deskripsi
-	-

b. Konstruktor

Atribut	Deskripsi
-	-

c. Selektor

Atribut	Deskripsi
-	-

d. Fungsi / Prosedur

Atribut	Deskripsi
public String toString()	Mengembalikan output berupa string: “No solution exists for the given system.”

3.1.6. LinearSystemSolution

Kelas ini merupakan kelas abstrak yang akan di-*extend* oleh kelas-kelas lainnya, yaitu UniqueSolution, ParametricSolution, dan NoSolution. Kelas ini hanya berfungsi sebagai penyatu ketiganya, sehingga tidak memiliki atribut, konstruktor, selektor dan metode.

3.2. Folder solvers

3.2.1. LinearSystemSolvers

a. Atribut

Atribut	Deskripsi
-	-

b. Konstruktor

Atribut	Deskripsi
-	-

c. Selektor

Atribut	Deskripsi
-	-

d. Fungsi / Prosedur

Atribut	Deskripsi
private boolean isAllZero(double[] row)	Mengecek apakah suatu baris berisi nol semua
private double[] subtractArray (double[] arr1, double[] arr2)	Mengurangi dua buah <i>array of double</i>
private double[] multArrayWithConst (double[] arr1, double k)	Mengalikan suatu array dengan suatu skalar
private boolean isNULL(double[] row)	Mengecek apakah suatu variabel (yang direpresentasikan dengan array) belum

	memiliki nilai
public LinearSystemSolution gaussianElimination (Matrix matrix)	Mengembalikan solusi sistem persamaan linear menggunakan metode eliminasi Gauss
public LinearSystemSolution gaussJordanElimination(Matrix matrix)	Mengembalikan solusi sistem persamaan linear menggunakan metode eliminasi Gauss-Jordan
public LinearSystemSolution cramersRule(Matrix matrix)	Mengembalikan solusi sistem persamaan linear menggunakan metode kaidah Cramer
public LinearSystemSolution inverseMethod(Matrix matrix)	Mengembalikan solusi sistem persamaan linear menggunakan metode matriks balikan

3.3. Folder mathmodels

3.3.1. Interpolation

a. Atribut

Atribut	Deskripsi
-	

b. Konstruktor

Atribut	Deskripsi
-	

c. Selektor

Atribut	Deskripsi

-	
---	--

d. Fungsi / Prosedur

Atribut	Deskripsi
public static PolynomialResult polynomialInterpolation(Matrix x, Matrix y)	Mengembalikan output berupa string: “No solution exists for the given system.”
public static Matrix getXInverseBicubicSpline ()	Mengembalikan balikan matriks X dalam persamaan untuk menentukan koefisien interpolasi <i>bicubic spline</i>
public static double bicubicSplineInterpolation (Matrix YInput, double a, double b, Matrix XInverse)	Mengembalikan hasil interpolasi <i>bicubic spline</i>

3.3.2. Regression

a. Atribut

Atribut	Deskripsi
-	

b. Konstruktor

Atribut	Deskripsi
-	

c. Selektor

Atribut	Deskripsi
-	

d. Fungsi / Prosedur

Atribut	Deskripsi
public static LinearSystemSolution multipleLinearRegression	Mengembalikan koefisien persamaan hasil regresi linear berganda
public static LinearSystemSolution multipleQuadraticRegression	Mengembalikan koefisien persamaan hasil regresi kuadratik berganda

3.4. Folder imageresizing

3.4.1. Image Resizing

a. Atribut

Atribut	Deskripsi
-	

b. Konstruktor

Atribut	Deskripsi
-	

c. Selektor

Atribut	Deskripsi
-	

-	
---	--

d. Fungsi / Prosedur

Atribut	Deskripsi
public static BufferedImage extendImage(BufferedImage original)	Memperbesar gambar asil dengan 2 pixel di atas, bawah, kiri, dan kanan agar interpolasi bisa menjangkau pixel yang lebih luas
public static BufferedImage resizeImage(BufferedImage original, double scaleX, double scaleY)	Mengubah ukuran gambar yang telah di- <i>extend</i>
private static Matrix[] getYInputForGroup(BufferedImage original, int baseX, int baseY)	Mempersiapkan YInput yaitu matriks 16x1 yang berisi, f , f_x , f_y , dan f_{xy} untuk masing-masing warna (<i>Red</i> , <i>Green</i> , <i>Blue</i>)
private static void interpolateRow(Matrix redChannel, Matrix greenChannel, Matrix blueChannel, int newY, Matrix[][] rowYInput, double[] normalizedXArray, double normalizedY, Matrix XInverse)	Menginterpolasi tiap pixel pada suatu baris pada matriks gambar yang baru
public static BufferedImage matrixToImage(Matrix[] rgbMatrices)	Mengubah matriks menjadi file gambar

3.5. Folder cli (Command Line Interface)

Folder ini merupakan program utama yang akan mengintegrasikan seluruh fitur pada aplikasi. Folder ini terdiri dari beberapa file menu dan submenu.

3.6. Folder gui (Graphical User Interface)

Folder ini merupakan pengembangan dari penerapan program utama. Pada folder ini, berisi file-file fxml untuk membuat layout dan elemen UI utama serta *styling*-nya. Selain itu, folder ini juga memuat file-file yang mengatur logika dari UI ketika diinteraksikan (diklik, diketik, dsb.)

Bab 4 : Eksperimen

4.1. Studi dan Tes Kasus

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

a. Soal:

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

Hasil eksperimen:

The screenshot shows a web-based linear system solver. At the top, there's a navigation bar with 'Hello!' and links to 'Home', 'Linear System Solver', and 'Gaussian Elimination Method'. The main title is 'Gaussian Elimination Method'. Below it, a note states: 'Gaussian elimination is a method for solving systems of linear equations by transforming the system's augmented matrix into a row echelon form (REF) using row operations. Once in REF, back substitution is used to find the solution for the system.' On the left, a 'Matrix' input field contains the following data:
1.00 1.00 -1.00 -1.00 1.00
2.00 5.00 -7.00 -5.00 -2.00
2.00 -1.00 1.00 3.00 4.00
5.00 2.00 -4.00 2.00 6.00
An 'Import from file' button is below this. In the center, a box displays 'Linear System Solution' with the message 'No Solution Found'. An arrow points from the matrix input to this box. To the right, another box labeled 'Explanation' contains the text 'No solution exists for the given system.' A 'Get Result' button is at the bottom right. A small note at the bottom says: '*Due to limitations of the "double" data type, precision is limited to two decimal places.'

Dari hasil eksperimen menggunakan eliminasi Gauss, dihasilkan bahwa persamaan tersebut tidak memiliki solusi.

b. Soal:

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

Hasil eksperimen:

Hello!

[Home](#) / [Linear System Solver](#) / [Gaussian Elimination Method](#)

Dark Mode

Gaussian Elimination Method

Gaussian elimination is a method for solving systems of linear equations by transforming the system's augmented matrix into a row echelon form (REF) using row operations. Once in REF, back substitution is used to find the solution for the system.

Matrix

```
1.00 -1.00 0.00 0.00 1.00 3.00
1.00 1.00 0.00 -3.00 0.00 6.00
2.00 -1.00 0.00 1.00 -1.00 5.00
-1.00 2.00 0.00 -2.00 -1.00 -1.00
```

->

[Import from file](#)

Linear System Solution

Parametric Solution Found

=

[Export to file](#)

Explanation

Parametric solution:
 $X_1 = 3.0 + a$
 $X_2 = 2.0a$
 $X_3 = b$
 $X_4 = -1.0 + a$
 $X_5 = a$

[Get Result](#)

*Due to limitations of the "double" data type, precision is limited to two decimal places.

Dari hasil eksperimen menggunakan metode eliminasi Gauss didapatkan solusi parametrik. Solusi terlampir pada gambar di atas.

c. Soal:

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

Hasil Eksperimen:

[Home](#) / [Linear System Solver](#) / [Gaussian Elimination Method](#)

Dark M

Gaussian Elimination Method

Gaussian elimination is a method for solving systems of linear equations by transforming the system's augmented matrix into a row echelon form (REF) using row operations. Once in REF, back substitution is used to find the solution for the system.

Matrix

```
0.00 1.00 0.00 0.00 1.00 0.00 2.00
0.00 0.00 0.00 1.00 1.00 0.00 -1.00
0.00 1.00 0.00 0.00 0.00 1.00 -1.00
```

->

[Import from file](#)

Linear System Solution

Parametric Solution Found

=

[Export to file](#)

Explanation

Parametric solution:
 $X_1 = c$
 $X_2 = 1.0 + -a$
 $X_3 = b$
 $X_4 = -2.0 + -a$
 $X_5 = 1.0 + a$
 $X_6 = a$

[Get Result](#)

*Due to limitations of the "double" data type, precision is limited to two decimal places.

Dari hasil eksperimen menggunakan metode eliminasi Gauss didapatkan hasil dengan solusi parametrik. Solusi terlampir pada gambar diatas.

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

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

[Home](#) / [Linear System Solver](#) / [Gauss Jordan Elimination Method](#)

Dark

Gauss Jordan Elimination Method

Gauss-Jordan elimination extends the Gaussian elimination method by transforming the matrix further into reduced row echelon form (RREF). This process simplifies the system to a point where each variable corresponds directly to a row in the matrix, making it easier to read off the solution without back substitution.

Matrix 1.00 0.50 0.33 0.25 0.20 0.67 1.00 0.50 0.33 0.25 0.20 0.67 0.14 0.00 0.33 0.25 0.20 0.67 0.14 0.13 0.00 0.25 0.20 0.67 0.14 0.13 0.11 0.00 0.20 0.67 0.14 0.13 0.11 0.10 0.00 0.67 0.14 0.13 0.11 0.10 0.09 0.00	->	Linear System Solution -0.26 0.35 -0.66 -0.18 -0.11 2.04	=	Explanation Unique solution: X1 = -0.259 X2 = 0.352 X3 = -0.657 X4 = -0.181 X5 = -0.107 X6 = 2.04
---	----	--	---	---

[Import from file](#) [Export to file](#) [Export to file](#) [Get Result](#)

*Due to limitations of the "double" data type, precision is limited to two decimal places.

Untuk $n=6$, hasilnya adalah parametrik

Hello!

[Home](#) / [Linear System Solver](#) / [Gaussian Elimination Method](#)

Dark Mode

Gaussian Elimination Method

Gaussian elimination is a method for solving systems of linear equations by transforming the system's augmented matrix into a row echelon form (REF) using row operations. Once in REF, back substitution is used to find the solution for the system.

Matrix 1.00 0.50 0.33 0.25 0.20 0.17 0.14 0.13 0.11 0.50 0.33 0.25 0.20 0.17 0.14 0.13 0.11 0.10 0.33 0.25 0.20 0.17 0.14 0.13 0.11 0.10 0.09 0.25 0.20 0.17 0.14 0.13 0.11 0.10 0.09 0.08 0.20 0.17 0.14 0.13 0.11 0.10 0.09 0.08 0.08 0.17 0.14 0.13 0.11 0.10 0.09 0.08 0.08 0.07 0.14 0.13 0.11 0.10 0.09 0.08 0.08 0.07 0.07 0.13 0.11 0.10 0.09 0.08 0.08 0.07 0.07 0.06 0.11 0.10 0.09 0.08 0.08 0.07 0.07 0.06 0.06 0.10 0.09 0.08 0.08 0.07 0.07 0.06 0.06 0.06	->	Linear System Solution 20.21 -167.09 340.14 -211.80 175.58 -66.15	=	Explanation Unique solution: X1 = 20.21310907803074 X2 = -167.08760735598713 X3 = 340.1436442229182 X4 = -211.79544173685275 X5 = 175.770890569415 X6 = -66.18889126224462 X7 = -284.9111005902079 X8 = -27.011188351999994 X9 = 89.32553599999999 X10 = 141.534
---	----	---	---	--

[Import from file](#) [Export to file](#) [Export to file](#) [Get Result](#)

*Due to limitations of the "double" data type, precision is limited to two decimal places.

Untuk $n=10$, didapatkan hasil parametrik.

2. SPL berbentuk matriks *augmented*

a. Soal:

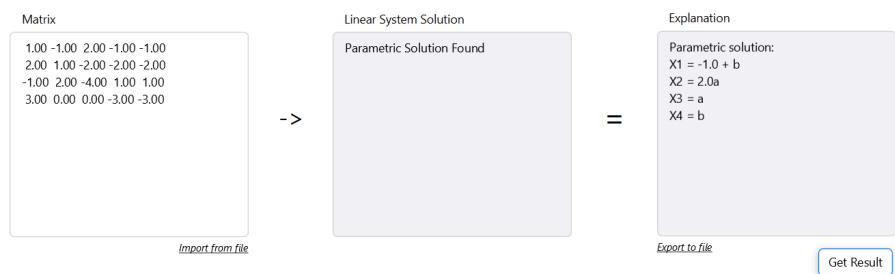
$$\begin{bmatrix} 1 & -1 & 2 & -1 & -1 \\ 2 & 1 & -2 & -2 & -2 \\ -1 & 2 & -4 & 1 & 1 \\ 3 & 0 & 0 & -3 & -3 \end{bmatrix}$$

Hasil Eksperimen:

[Home](#) / [Linear System Solver](#) / [Gauss Jordan Elimination Method](#)

Gauss Jordan Elimination Method

Gauss-Jordan elimination extends the Gaussian elimination method by transforming the matrix further into reduced row echelon form (RREF). This process simplifies the system to a point where each variable corresponds directly to a row in the matrix, making it easier to read off the solution without back substitution.



*Due to limitations of the "double" data type, precision is limited to two decimal places.

Dari hasil persamaan menggunakan metode eliminasi Gauss-Jordan didapatkan parametrik terlampir pada gambar di atas.

b. Soal:

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

Hasil eksperimen:

[Home](#) / [Linear System Solver](#) / [Gaussian Elimination Method](#)

Gaussian Elimination Method

Gaussian elimination is a method for solving systems of linear equations by transforming the system's augmented matrix into a row echelon form (REF) using row operations. Once in REF, back substitution is used to find the solution for the system.



*Due to limitations of the "double" data type, precision is limited to two decimal places.

Dari hasil persamaan menggunakan metode eliminasi Gauss didapatkan solusi unik terlampir pada gambar di atas.

3. SPL berbentuk

a. Soal:

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

Hasil Eksperimen:

[Home](#) / [Linear System Solver](#) / [Gauss Jordan Elimination Method](#)

Dark Mode

Cramer's Rule Method

Cramer's rule solves a system of linear equations using determinants. It applies to square matrices with a non-zero determinant. Each variable is found by replacing its corresponding column in the matrix with the constants, then dividing the determinant of the modified matrix by the determinant of the original matrix.

Matrix	Linear System Solution	Explanation
<pre> 8.00 1.00 3.00 2.00 0.00 2.00 9.00 -1.00 -2.00 1.00 1.00 3.00 2.00 -1.00 2.00 1.00 0.00 6.00 4.00 3.00 </pre>	<p>-></p> <div style="border: 1px solid black; padding: 5px; width: fit-content; margin: auto;"> $-0.22 \ 0.18 \ 0.71 \ -0.26$ </div>	<p>=</p> <div style="border: 1px solid black; padding: 5px; width: fit-content; margin: auto;"> <p>Unique solution: $X_1 = -0.2243243243243243$ $X_2 = 0.1824324324324324$ $X_3 = 0.7094594594594594$ $X_4 = -0.2581081081081081$</p> </div>
Import from file	Export to file	Export to file
<input type="button" value="Get Result"/>		

*Due to limitations of the "double" data type, precision is limited to two decimal places.

Dari hasil persamaan menggunakan metode Cramer diidapatkan solusi unik terlampir pada gambar di atas.

b.

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

Gaussian Elimination Method

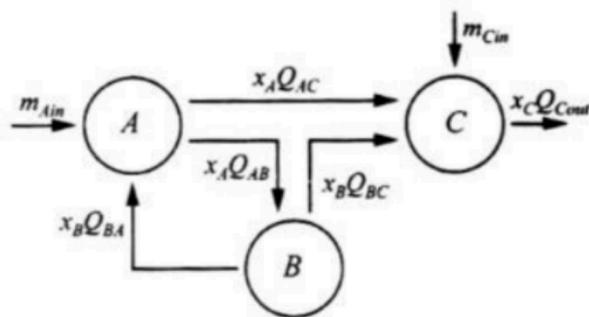
Gaussian elimination is a method for solving systems of linear equations by transforming the system's augmented matrix into a row echelon form (REF) using row operations. Once in REF, back substitution is used to find the solution for the system.

Matrix	Linear System Solution	Explanation
<pre> 0.00 0.00 0.00 1.00 1.00 1.00 0.00 0.00 0.00 1.00 1.00 1.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.04 0.00 0.04 0.75 0.04 0.75 0.6 0.00 0.25 0.91 0.25 0.91 0.25 0.91 0.25 0.0 0.61 0.75 0.04 0.75 0.04 0.00 0.04 0.00 0.0 0.00 0.00 1.00 0.00 0.00 1.00 0.00 0.00 1.0 0.00 1.00 0.00 0.00 1.00 0.00 0.00 1.00 0.0 1.00 0.00 0.00 1.00 0.00 0.00 1.00 0.00 0.0 0.04 0.75 0.61 0.00 0.04 0.75 0.00 0.00 0.4 0.91 0.25 0.00 0.25 0.91 0.25 0.00 0.25 0.9 0.04 0.00 0.00 0.75 0.04 0.00 0.61 0.75 0.0 </pre>	No Solution Found	No solution exists for the given system.
<i>Import from file</i>	<i>Export to file</i>	<i>Get Result</i>

*Due to limitations of the "double" data type, precision is limited to two decimal places.

Dari hasil eksperimen menggunakan metode eliminasi Gauss, hasil persamaan diatas tidak memiliki solusi.

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:

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

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

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

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 $Q_{C_{out}} = 150 \text{ } m^3/s$ dan $m_{A_{in}} = 1300$ dan $m_{C_{in}} = 200 \text{ } mg/s$.

Hasil eksperimen:

Gauss Jordan Elimination Method

Gauss-Jordan elimination extends the Gaussian elimination method by transforming the matrix further into reduced row echelon form (RREF). This process simplifies the system to a point where each variable corresponds directly to a row in the matrix, making it easier to read off the solution without back substitution.

Matrix <pre>-12.00 60.00 0.00 -1300.00 40.00 -80.00 0.00 0.00 80.00 20.00 -150.00 -200.00</pre>	->	Linear System Solution <pre>14.44 7.22 10.00</pre>	=	Explanation Unique solution: X1 = 14.44 X2 = 7.222 X3 = 10.0
Import from file		Export to file		Export to file
				Get Result

*Due to limitations of the "double" data type, precision is limited to two decimal places.

Dari hasil eksperimen menggunakan metode Gauss-Jordan didapatkan solusi dengan hasil unik.

5. Studi Kasus Interpolasi

- 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:

$$\begin{array}{ll} x = 0.2 & f(x) = -3.036 \\ x = 0.55 & f(x) = -0.047 \\ x = 0.85 & f(x) = -0.023 \\ x = 1.28 & f(x) = -0.047 \end{array}$$

Hasil Eksperimen:

X Value	Interpolated Y
<input style="width: 20px; border: 1px solid #ccc; border-radius: 5px; padding: 2px;" type="text" value="X"/> 0.2	<input style="width: 20px; border: 1px solid #ccc; border-radius: 5px; padding: 2px;" type="text" value="Y"/> -0.023
Get Interpolated	

X Value	Interpolated Y
<input style="width: 20px; border: 1px solid #ccc; border-radius: 5px; padding: 2px;" type="text" value="X"/> 0.55	<input style="width: 20px; border: 1px solid #ccc; border-radius: 5px; padding: 2px;" type="text" value="Y"/> -0.047
Get Interpolated	

X Value		Interpolated Y	
X	0.85	Y	-0.023
<input type="button" value="Get Interpolated"/>			

X Value		Interpolated Y	
X	1.28	Y	-0.047
<input type="button" value="Get Interpolated"/>			

- 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:

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

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

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

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

- a. 16/07/2022
- 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.

The screenshot shows a web-based application for polynomial interpolation. On the left, a 'Points' section lists data points: 6.57 12.62, 7.00 21.81, 7.26 38.39, 7.45 54.52, 7.55 51.95, 7.84 28.23, 8.16 35.76, 8.48 20.81, 8.71 12.41, 9.00 10.53. An arrow points to the 'Interpolation Result' section, which displays the formula $f(x) = 7.97907171721E9 + -1.02957660048x$. Below this is an 'X Value' input field set to 7.516 and an 'Interpolated Y' output field showing -1530128.36753. A 'Get Interpolated' button is present. At the bottom, there are 'Import from file' and 'Export to file' buttons, and a note: "Due to limitations of the 'double' data type, precision is limited to two decimal places."

This screenshot shows the same application with different input values. The 'Points' section remains the same. The 'Interpolation Result' section now displays the formula $1.75693711088E9x^3 + 3.68569416138E8x^2$. The 'X Value' input field is set to 9.1667, and the 'Interpolated Y' output field shows -1296850.65570. A 'Get Interpolated' button is visible. A note at the bottom states: "Due to limitations of the 'double' data type, precision is limited to two decimal places."

This screenshot shows the application again with different input values. The 'Points' section remains the same. The 'Interpolation Result' section displays the formula $f(x) = 7.97907171721E9 + -1.02957660048x$. The 'X Value' input field is set to 7.9032, and the 'Interpolated Y' output field shows -1569825.78159. A 'Get Interpolated' button is visible. A note at the bottom states: "Due to limitations of the 'double' data type, precision is limited to two decimal places."

Berdasarkan hasil eksperimen diatas didapatkan 3 nilai, yakni

1. Untuk 16/07/2022 mendapatkan hasil -1530128.3675
 2. Untuk 10/08/2022 mendapatkan hasil -1723858.3835
 3. Untuk 05/09/2022 mendapatkan hasil -1569825.7815
- 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 $h = (2 - 0)/5 = 0.4$.

Hasil Eksperimen:

Dari fungsi tersebut didapatkan nilai

Untuk $f(0) = 0$
 Untuk $f(0.4) = 0.4189$
 Untuk $f(0.8) = 0.5072$
 Untuk $f(1.2) = 0.5609$
 Untuk $f(1.6) = 0.5837$
 Untuk $f(2.0) = 0.5767$

Dari kelima titik tersebut dapat didapatkan persamaan polinomial sebagai berikut:
 $f(x) = 0.0 + 2.034x^1 + -3.549x^2 + 3.231x^3 + -1.418x^4 + 0.236x^5$

Misalkan ingin mengestimasi nilai 1,15:

Didapatkan nilai :0.5540

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

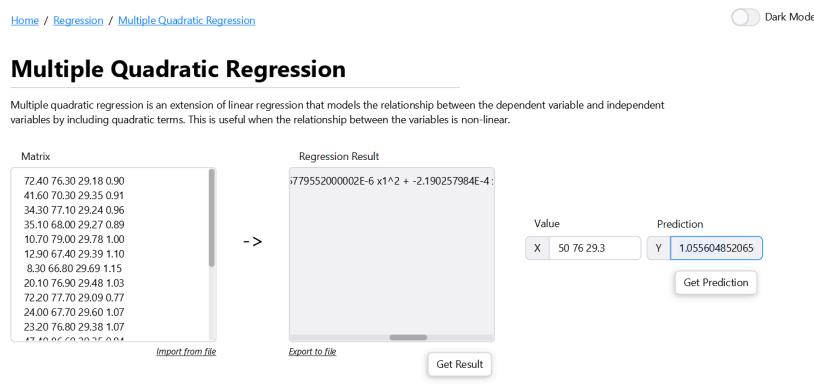
Source: Charles T. Hare, "Light-Duty Diesel Emission Correction Factors for Ambient Conditions," EPA-600/2-77-116. U.S. Environmental Protection Agency.

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.

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

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



Dari hasil eksperimen didapatkan nilai hasil prediksi menggunakan multiple quadratic regression dengan nilai 1.0556.

Multiple Linear Regression

Multiple linear regression is a statistical technique used to predict the value of a dependent variable based on multiple independent variables. The relationship is modeled as a linear equation where coefficients are estimated to minimize the difference between predicted and actual values.

Matrix

```
72.40 76.30 29.18 0.90
41.60 70.30 29.35 0.91
34.30 77.10 29.24 0.96
35.10 68.00 29.27 0.89
10.70 79.00 29.78 1.00
12.90 67.40 29.39 1.10
8.30 66.80 29.69 1.15
20.10 76.90 29.48 1.03
72.20 77.70 29.09 0.77
24.00 67.70 29.60 1.07
23.20 76.80 29.38 1.07
47.40 80.60 29.35 0.88
```

[Import from file](#)

Regression Result

```
f(x) = -3.459 + -0.002x1 + 0.0x2 +
0.154x3
```

Value Prediction

X	50 76 29.3	Y	0.953199999999
---	------------	---	----------------

[Get Prediction](#)

[Export to file](#)

[Get Result](#)

*Due to limitations of the "double" data type, precision is limited to two decimal places.

Dari hasil eksperimen didapatkan nilai hasil prediksi menggunakan multiple linear regression dengan nilai 0.952.

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:

$$f(0, 0) = 21$$

```
1. Polynomial Interpolation
2. Bicubic Spline Interpolation
3. Back to main menu
-> Select an option (1-3): 2

-> Matrix should be 4 x 4:
-> Input from file or from CLI?
1. Input Matrix from File
2. Input Matrix from CLI
2

-> Enter each element of the matrix:
21 98 125 153
51 101 161 59
0 42 72 210
16 12 81 96

-> Enter x value for point to be interpolated : 0

-> Enter y value for point to be interpolated : 0

Fitting the data...21.0

Result:
21.0
```

$$f(0.5, 0.5) = 87.79$$

```

1. Polynomial Interpolation
2. Bicubic Spline Interpolation
3. Back to main menu
-> Select an option (1-3): 2

-> Matrix should be 4 x 4:
-> Input from file or from CLI?
1. Input Matrix from File
2. Input Matrix from CLI
2

-> Enter each element of the matrix:
21 98 125 153
51 101 161 59
0 42 72 210
16 12 81 96

-> Enter x value for point to be interpolated : 0.5
-> Enter y value for point to be interpolated : 0.5

Fitting the data...87.796875

Result:
87.796875

```

$$f(0.25, 0.75) = 82.148$$

```

1. Polynomial Interpolation
2. Bicubic Spline Interpolation
3. Back to main menu
-> Select an option (1-3): 2

-> Matrix should be 4 x 4:
-> Input from file or from CLI?
1. Input Matrix from File
2. Input Matrix from CLI
2

-> Enter each element of the matrix:
21 98 125 153
51 101 161 59
0 42 72 210
16 12 81 96

-> Enter x value for point to be interpolated : 0.25
-> Enter y value for point to be interpolated : 0.75

Fitting the data...82.148193359375

Result:
82.148193359375

```

$$f(0.1, 0.9) = 91.271$$

```

1. Polynomial Interpolation
2. Bicubic Spline Interpolation
3. Back to main menu
-> Select an option (1-3): 2

-> Matrix should be 4 x 4:
-> Input from file or from CLI?
1. Input Matrix from File
2. Input Matrix from CLI
2

-> Enter each element of the matrix:
21 98 125 153
51 101 161 59
0 42 72 210
16 12 81 96

-> Enter x value for point to be interpolated : 0.1

-> Enter y value for point to be interpolated : 0.9

Fitting the data...91.27126700000001

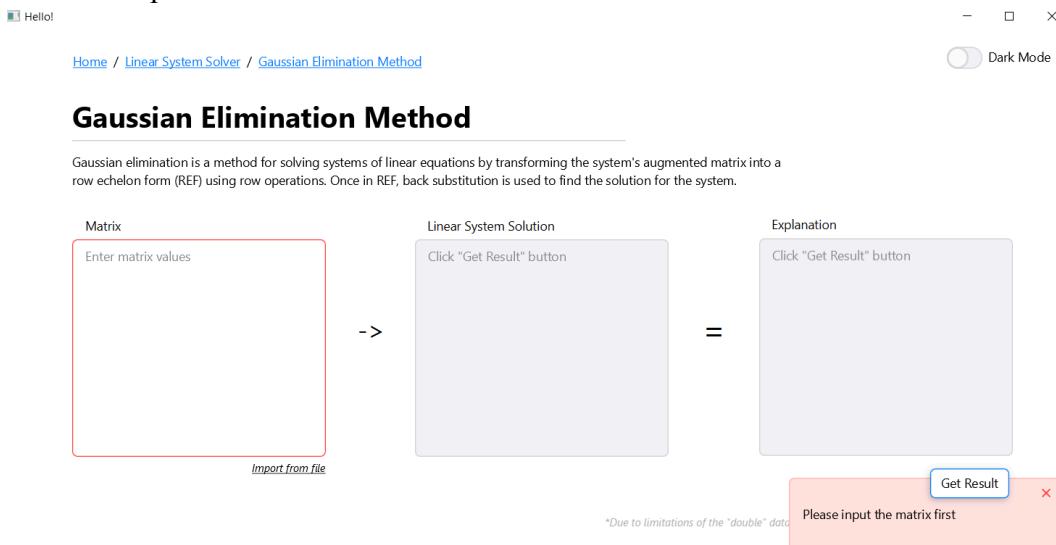
Result:
91.27126700000001

```

Analisis:

Hasil pada gui mengalami keterbatasan karena input akan di-round 2 angka di belakang koma. oleh karena itu, hasil yang lebih akurat didapat pada command line interface yang juga telah dirilis.

8. Studi Kasus Input Invalid



Jika pengguna tidak memberikan input, sistem akan memberi notifikasi untuk memperingatkan mengisi input.

Inverse Method

The inverse method for solving a system of linear equations involves finding the inverse of the coefficient matrix. If the inverse exists, the solution to the system $Ax=b$ is given by $x=A^{-1}b$, where A is the coefficient matrix, x is the vector of unknowns, and b is the vector of constants.

Matrix

```
1.00 2.00 3.00 4.00  
1.00 2.00 3.00 4.00  
1.00 2.00 3.00 4.00
```

Linear System Solution

Calculating...

Explanation

Calculating...

->

=

[Import from file](#)[Get Result](#)*Due to limitations of the "double" data type, the result may not be exact.

Linear system solver failed.

Untuk beberapa kasus (seperti metode inverse) memiliki beberapa persyaratan tertentu. Untuk kasus ini diperlukan matriks X berukuran persegi dan perlu memiliki Determinan. Jika kedua syarat tersebut tidak memenuhi, maka sistem akan memberikan peringatan.

Bab 5 : Kesimpulan, Saran, dan Refleksif

5.1. Kesimpulan

Aljabar linear merupakan bidang yang sangat dekat dengan pemrograman. Inti dari tugas kali ini adalah menyelesaikan persamaan linear menggunakan suatu langkah-langkah yang pasti dan terukur. Oleh karena itulah, persoalan ini dapat diselesaikan dengan sebuah program. Ketika kita berhasil menyelesaikan suatu sistem persamaan linear, kita dapat menyelesaikan masalah-masalah lain yang lebih kompleks. seperti interpolasi polinomial, regresi linear, regresi kuadratik yang memiliki kemampuan untuk melakukan prediksi. Bahkan, aplikasi dari aljabar linear bisa digunakan untuk memproses citra suatu gambar dengan mengubah ukuran-ukuran pixelnya menggunakan suatu teknik yang disebut bicubic spline interpolation. Kesimpulannya, aljabar linear khususnya sistem persamaan linear memiliki cakupan dan aplikasi yang sangat luas, terutama di bidang komputasi.

5.2. Saran

Saran dari kelompok kami kepada para asisten adalah agar lebih spesifik dan jelas dalam memberikan studi kasus. Terutama, setidaknya diberi satu atau dua luaran agar kami lebih yakin bahwa program yang kami buat sudah cukup berhasil.

5.3. Komentar

Menurut kami, tugas ini sangat seru dan mempertajam kemampuan kami dalam menerjemahkan masalah nyata ke dalam permasalahan aljabar linear, lalu mentranslasikannya dalam kode. Selain itu, pemilihan bahasa pemrograman Java menurut kami sangat tepat karena kami langsung dapat menerapkan hal yang sedang dipelajari juga di mata kuliah algoritma dan struktur data, yaitu *Abstract Data Type* (ADT) atau tipe bentukan. Selain itu, dengan menggunakan bahasa Java, kami juga jadi punya gambaran mengenai paradigma *Object Oriented Programming* (ADT)

5.4. Refleksi

Jangan terlena dengan tugas lain dan tetap kerjakan sesegera mungkin walaupun sudah hampir selesai. Overall, perencanaan sudah sangat baik, metode testing sudah sangat baik, dan pembagian tugas juga sudah sangat baik.