IF2123 ALJABAR LINEAR DAN GEOMETRI

IMPLEMENTASI SISTEM LINEAR, REGRESI, DAN INTERPOLASI DENGAN BAHASA PEMROGRAMAN JAVA

Laporan Tugas Besar 1

Disusun untuk memenuhi tugas mata kuliah Aljabar Linear dan Geometri pada Semester 1 (satu)

Tahun Akademik 2024/2025

Oleh:

Rhio Bimo Prakoso S	**13523125**
Brian Ricardo Tamin	**13523126**
Arlow Emanuel Hergara	**13523161**

Kelompok Janji Jawa

PROGRAM STUDI TEKNIK INFORMATIKA

SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA

INSTITUT TEKNOLOGI BANDUNG

BANDUNG

2024

DAFTAR ISI

DAFTAR ISI **2**
BAB I **4**
1.1 Latar Belakang Masalah 4
1.2 Spesifikasi Program 4
BAB II **6**
2.1 Sistem Persamaan Linear 6
2.2 Matriks Invers 8
2.3 Interpolasi 9
2.4 Regresi 10
2.5 Matriks Adjoin 11
2.6 Determinan 11
BAB III **13**
3.1 Class Matrix 13
3.2 Class DeterminantCofactorSolver 15
3.3 Class DeterminantOBESolver 16
3.4 Class BicubicSplineInterpolator 17
3.5 Class ImageResizer 19
3.6 Class PolynomialInterpolator 20
3.7 Class InverseNormalSolver 22
3.8 Class InverseOBESolver 23
3.9 Class CramerSolver 24
3.10 Class GaussJordanSolver 25
3.11 Class GaussSolver 26
3.12 Class InverseSolver 27
3.13 Class OBEAdder 27
3.14 Class OBEScaler 28
3.15 Class OBESwitcher 28
3.16 Class LinearRegressor 29
3.17 Class QuadraticRegressor 30
BAB IV **32**
4.1 Studi Kasus #1 (Temukan solusi SPL, Ax = b) 32
4.2 Studi Kasus #2 (SPL Matriks Augmented) 36
4.3 Studi Kasus #3 (SPL berbentuk) 38
4.4 Studi Kasus #4 (Sistem Reaktor) 41
4.5 Studi Kasus #5 (Interpolasi) 42
4.6 Studi Kasus #6 (Linear Kuadratik Berganda) 49
4.7 Studi Kasus #7 (Interpolasi Bicubic Spline) 51
BAB V **55**
5.1 Kesimpulan 55
5.2 Komentar 55

5.3 Refleksi.....56
5.4 Saran.....56
LAMPIRAN.....58
Image Resizer..... 58

BAB I

DESKRIPSI MASALAH

1.1 Latar Belakang Masalah

Dalam mata kuliah Aljabar Linear dan Geometri, pemahaman konsep matriks dan operasinya sangat penting untuk memecahkan berbagai permasalahan matematis dan aplikatif. Matriks merupakan alat fundamental dalam banyak disiplin ilmu, termasuk teknik, ekonomi, dan ilmu komputer, karena kemampuannya untuk menyajikan dan menyelesaikan sistem persamaan linear, menghitung determinan, serta melakukan operasi lainnya.

Untuk memenuhi Tugas Besar 1, kami mengembangkan sebuah program aljabar linear menggunakan bahasa pemrograman Java. Program ini dirancang untuk menyelesaikan berbagai permasalahan yang berkaitan dengan matriks dengan cara yang efisien dan user-friendly. Fitur-fitur yang disediakan dalam program ini meliputi:

1. Sistem Persamaan Linear: Menyediakan solusi untuk sistem persamaan linear dengan berbagai metode.
2. Determinasi: Menghitung nilai determinan dari matriks.
3. Matriks Invers: Menghitung invers dari matriks yang memenuhi syarat.
4. Interpolasi: Menyediakan metode interpolasi untuk analisis data.
5. Regresi: Memungkinkan pengguna untuk melakukan analisis regresi untuk memodelkan hubungan antara variabel.

Dengan pengembangan program ini, diharapkan pengguna dapat memahami dan menerapkan konsep aljabar linear dengan lebih baik, serta mampu menyelesaikan permasalahan yang lebih kompleks dengan bantuan teknologi. Kami percaya bahwa penguasaan alat ini akan memberikan kontribusi signifikan dalam belajar dan aplikasi ilmu aljabar linear.

1.2 Spesifikasi Program

Program ini dirancang dengan antarmuka berbasis teks yang mudah digunakan, memungkinkan pengguna untuk memasukkan data secara interaktif. Setiap fitur dalam program diimplementasikan dengan algoritma yang sesuai untuk menangani berbagai ukuran matriks dan kompleksitas sistem persamaan. Dengan demikian, program ini tidak hanya mampu memberikan solusi numerik yang akurat, tetapi juga dapat membantu menganalisis pola data dan fungsi dengan pendekatan matematis yang beragam.

1.2.1 Bahasa Pemrograman

Bahasa pemrograman yang digunakan dalam pengembangan tugas besar ini adalah Java. Pemilihan Java didasarkan pada kemampuannya untuk menangani struktur data yang kompleks serta dukungan yang kuat terhadap konsep pemrograman berorientasi objek. Selain itu, Java terkenal akan stabilitas dan portabilitasnya, yang memungkinkan kode yang ditulis untuk dijalankan di berbagai platform tanpa perlu melakukan modifikasi signifikan. Dengan demikian, penggunaan Java dalam proyek ini diharapkan dapat mempermudah pengembangan dan pemeliharaan program aljabar linear.

1.2.2 GUI (*Graphical User Interface*)

Dengan menggunakan library Swing dalam Java untuk membuat GUI (Graphical User Interface), diharapkan pengguna dapat lebih mudah memahami dan lebih cepat mengerti dalam menggunakan program yang dibuat. GUI ini juga berfungsi untuk membuat proses input/output menjadi lebih jelas.

Bentuk GUI yang diprogram terdiri dari pilihan menu dengan sub-menu yang relevan jika diperlukan (contoh: Sistem Persamaan Linear (SPL) → Metode Eliminasi Gauss). Selain itu, disediakan interface untuk memasukkan nilai-nilai dari persoalan yang dihadapi, sehingga pengguna dapat berinteraksi dengan program dengan lebih intuitif.

BAB II

TEORI SINGKAT

2.1 Sistem Persamaan Linear

Sistem persamaan linear adalah kumpulan dua atau lebih persamaan linear yang melibatkan variabel yang sama. Sistem ini dapat diselesaikan untuk menemukan nilai-nilai variabel yang memenuhi semua persamaan secara simultan.

2.1.1 Metode Eliminasi Gauss

Metode eliminasi Gauss adalah teknik untuk menyelesaikan sistem persamaan linear dengan mengubah sistem menjadi bentuk segitiga atas menggunakan operasi baris elementer. Langkah-langkahnya meliputi:

1. Eliminasi: Menggunakan operasi baris untuk mengeliminasi variabel satu per satu.
2. Substitusi Maju: Setelah sistem berada dalam bentuk segitiga atas, mulai dari persamaan terakhir untuk menentukan nilai variabel.

Contoh:

$$\begin{cases} 2x + 3y = 8 \\ 5x - y = 2 \end{cases}$$

Langkah eliminasi Gauss:

1. Eliminasi variabel x

   Mengalikan persamaan pertama dengan 5 dan persamaan kedua dengan 2:

   $$\begin{cases} 10x + 15y = 40 \\ 10x - 2y = 4 \end{cases}$$

   Mengurangkan persamaan kedua dari persamaan pertama:

   $$17y = 36 \Rightarrow y = \frac{36}{17}$$

2. Substitusi nilai y ke persamaan kedua:

   $$5x - \frac{36}{17} = 2 \Rightarrow 5x = \frac{70}{17} \Rightarrow x = \frac{14}{17}$$

2.1.2 Metode Eliminasi Gauss-Jordan

Metode eliminasi Gauss-Jordan adalah pengembangan dari metode eliminasi Gauss yang tidak hanya mengubah sistem menjadi bentuk segitiga atas tetapi juga menjadi bentuk baris eselon tereduksi. Dengan demikian, solusi dapat diperoleh langsung tanpa perlu substitusi maju. Langkah-langkahnya meliputi:

1. Eliminasi: Seperti pada metode Gauss, gunakan operasi baris untuk mengubah matriks augmented menjadi bentuk segitiga.
2. Normalisasi: Membuat setiap pivot (elemen utama) menjadi 1 dan mengeliminasi variabel di atas pivot.

Contoh:

$$\begin{cases} 2x + 3y = 8 \\ 5x - y = 2 \end{cases}$$

Setelah eliminasi Gauss-Jordan, matriks augmented menjadi:

$$\begin{pmatrix} 1 & 0 & \frac{14}{17} \\ 0 & 1 & \frac{36}{17} \end{pmatrix}$$

Sehingga solusi adalah $x = \frac{14}{17}$ dan $y = \frac{36}{17}$.

2.1.3 Metode Cramer

Metode Cramer adalah teknik penyelesaian sistem persamaan linear yang menggunakan determinan matriks. Metode ini hanya berlaku untuk sistem yang memiliki jumlah persamaan sama dengan jumlah variabel dan determinan matriks koefisien tidak nol.

Untuk sistem:

$$Ax = b$$

Solusi untuk setiap variabel x_i diberikan oleh:

$$x_i = \frac{det(A_i)}{det(A)}$$

dimana A_i adalah matriks yang diperoleh dengan mengganti kolom ke-i dari A dengan vektor b.

2.1.4 Metode Invers

Metode invers menggunakan matriks invers untuk menyelesaikan sistem persamaan linear. Jika A adalah matriks koefisien dan A^{-1} adalah inversnya, maka solusi x diberikan oleh :

$$x = A^{-1}b$$

Langkah-langkah:

1. Hitung invers matriks A.
2. Kalikan invers tersebut dengan vektor b untuk mendapatkan solusi.

Untuk sistem:

$$\mathrm{A} = \begin{pmatrix} 2 & 3 \\ 5 & -1 \end{pmatrix}, \mathrm{b} = \begin{pmatrix} 8 \\ 2 \end{pmatrix}$$

Invers matriks A:

$$\mathrm{A}^{-1} = \frac{1}{\det(\mathrm{A})}\begin{pmatrix} -1 & -3 \\ -5 & 2 \end{pmatrix} = \frac{1}{-17}\begin{pmatrix} -1 & -3 \\ -5 & 2 \end{pmatrix} = \begin{pmatrix} \frac{1}{17} & \frac{3}{17} \\ \frac{5}{17} & -\frac{2}{17} \end{pmatrix}$$

Solusi :

$$\mathrm{x} = \mathrm{A}^{-1}\mathrm{b} = \begin{pmatrix} \frac{1}{17} & \frac{3}{17} \\ \frac{5}{17} & -\frac{2}{17} \end{pmatrix} = \begin{pmatrix} 8 \\ 2 \end{pmatrix} = \begin{pmatrix} \frac{14}{17} \\ \frac{36}{17} \end{pmatrix}$$

2.2 Matriks Invers

Matriks invers adalah matriks yang, jika dikalikan dengan matriks aslinya, menghasilkan matriks identitas. Matriks invers hanya ada untuk matriks persegi yang memiliki determinan tidak nol.

$$AA^{-1} = A^{-1}A = I_n$$

2.2.1 Metode Normal

Metode normal untuk menemukan invers matriks melibatkan penggunaan rumus invers untuk matriks 2x2 dan 3x3. Untuk matriks berordo lebih besar, metode ini menjadi lebih kompleks dan seringkali tidak praktis tanpa bantuan komputer.

Contoh Matriks 2x2:

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

2.2.2 Metode Operasi Baris Elementer (OBE)

Metode Operasi Baris Elementer (OBE) adalah teknik yang lebih umum dan efisien untuk menemukan invers matriks, terutama untuk matriks berordo lebih besar. Langkah-langkah meliputi:

1. Matriks Augmented: Gabungan matriks A dengan matriks identitas I_n:

$$[A|I_n]$$

2. Eliminasi: Gunakan operasi baris elementer untuk mengubah A menjadi .
3. Invers: setelah A menjadi I_n matriks yang tadinya I_n berubah menjadi A^{-1}.

$$A = \begin{pmatrix} 2 & 3 \\ 5 & -1 \end{pmatrix}, I_2 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

Matriks augmented:

$$\left[\begin{array}{cc|cc} 2 & 3 & 1 & 0 \\ 5 & -1 & 0 & 1 \end{array}\right]$$

2.3 Interpolasi

Interpolasi adalah metode untuk memperkirakan nilai di antara dua titik data yang diketahui. Dalam konteks ini, interpolasi digunakan untuk membangun fungsi yang melewati titik-titik data tersebut.

2.3.1 Interpolasi Polinom

Interpolasi polinom adalah metode untuk menentukan polinom dari derajat $n - 1$ yang melewati n titik data. Polinom interpolasi dapat digunakan untuk memperkirakan nilai fungsi di antara titik-titik yang diketahui.

2.3.2 Interpolasi Bicubic Spline

Interpolasi bicubic spline adalah metode interpolasi yang menggunakan polinom kubik dalam dua variabel untuk menghasilkan permukaan yang halus dan kontinu. Metode ini sering digunakan dalam aplikasi grafis dan pemrosesan citra untuk menghasilkan transisi yang mulus antara titik-titik data.

Rumus Umum:

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

Koefisien a_{ij} ditentukan berdasarkan kondisi kekontinuan dan derivatif pada titik-titik grid yang diketahui. Interpolasi bicubic spline lebih kompleks dibandingkan interpolasi polinomial satu variabel, namun memberikan hasil yang lebih halus dan akurat pada data dua dimensi.

2.4 Regresi

Regresi adalah teknik statistik yang digunakan untuk memodelkan dan menganalisis hubungan antara satu variabel dependen dengan satu atau lebih variabel independen.

2.4.1 Regresi Linear Berganda

Regresi linear berganda adalah metode untuk memodelkan hubungan antara satu variabel dependen dan dua atau lebih variabel independen. Model ini diasumsikan memiliki bentuk linear.

Model Regresi Linear Berganda :

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_n x_n + \epsilon$$

- y adalah variabel dependen
- $x_1, x_2, \ldots, x_n$ adalah variabel independen
- $\beta_0, \beta_1, \ldots, \beta_n$ adalah koefisien regresi
- ε adalah error term.

2.4.2 Regresi Kuadratik Berganda

Regresi kuadratik berganda adalah perluasan dari regresi linear berganda yang memasukkan istilah kuadrat dari variabel independen untuk menangkap hubungan non-linear antara variabel dependen dan independen.

Model Regresi Kuadratik Berganda:

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

2.5 Matriks Adjoin

Matriks adjoin (adjoint matrix) adalah matriks yang diperoleh dari matriks kofaktor, di transposisikan. Matriks adjoin digunakan dalam perhitungan invers matriks dan determinan.

Definisi:
Jika A adalah matriks berordo $n \; x \; n$, maka matriks adjoin $adj(A)$ adalah:

$$adj(A) \; = \; C^T$$

di mana C adalah matriks kofaktor dari A.

Contoh:

$$\mathrm{A} = \begin{pmatrix} \mathrm{a} & \mathrm{b} \\ \mathrm{c} & \mathrm{d} \end{pmatrix}, \qquad \mathrm{adj(A)} = \begin{pmatrix} \mathrm{d} & \text{-b} \\ \text{-c} & \mathrm{a} \end{pmatrix}$$

2.6 Determinan

Determinan adalah nilai skalar yang dapat dihitung dari matriks persegi dan memiliki peran penting dalam menentukan sifat-sifat matriks, seperti keberadaan invers dan solusi dari sistem persamaan linear.

2.6.1 Metode Kofaktor

Metode kofaktor adalah teknik untuk menghitung determinan matriks dengan mengembangkan determinan melalui ekspansi kofaktor.

$$\mathbf{A} = \begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix}, \quad \det(\mathbf{A}) = a(ei - fh) - b(di - fg) + c(dh - eg)$$

Untuk matriks $n\ x\ n$, determinan dapat dihitung dengan memilih satu baris atau kolom dan mengembangkan melalui kofaktor.

$$\det(\mathbf{A}) = \sum_{j=1}^{n} (-1)^{i+j} a_{ij} \det(\mathbf{A}_{ij})$$

di mana A_{ij} adalah minor yang diperoleh dengan menghapus baris i dan kolom j dari A.

2.6.2 Metode Operasi Baris Elementer (OBE)

Metode OBE untuk menghitung determinan melibatkan transformasi matriks ke bentuk segitiga atas dan kemudian mengalikan elemen diagonal utama.

Contoh:

$$\mathbf{A} = \begin{pmatrix} 2 & 3 \\ 5 & -1 \end{pmatrix} \quad R2 = R2 - \frac{5}{2}R1 \Rightarrow \begin{pmatrix} 2 & 3 \\ 0 & -\frac{17}{2} \end{pmatrix}$$

determinan :

$$\det(\mathbf{A}) = 2 \times \left(-\frac{17}{2}\right) = -17$$

BAB III

IMPLEMENTASI PROGRAM

3.1 Class Matrix

3.1.1. Atribut

Nama Atribut	Penjelasan Singkat
data	Matriks dua dimensi bertipe double yang menyimpan elemen-elemen matriks.

3.1.2. Metode

3.1.2.1. Utility Operations

Nama Metode	Garis Besar Cara Kerja
GetIdentity(int size)	Menghasilkan matriks identitas berukuran size x size. Setiap elemen diagonal bernilai 1.
Append(Matrix A, Matrix B)	Menggabungkan dua Matriks A dan B secara horizontal, dengan syarat jumlah baris keduanya sama.
GetMinor(int i , int j)	Menghasilkan submatriks dengan menghilangkan baris ke-i dan kolom ke-j dari matriks. Digunakan untuk perhitungan determinan.

3.1.2.2. Basic Operations

Nama Metode	Garis Besar Cara Kerja
Add(Matrix A, Matrix B)	Menjumlahkan dua matriks dengan ukuran yang sama, menghasilkan matriks baru sebagai hasil penjumlahan elemen-elemennya.
Subtract(Matrix A, Matrix B)	Mengurangi matriks A dengan B

	dengan ukuran yang sama, menghasilkan matriks baru sebagai hasil pengurangan elemen-elemennya.
Multiply(int i , int j)	Mengalikan dua matriks jika jumlah kolom A sama dengan jumlah baris B, menghasilkan matriks baru sebagai hasil perkalian.
Multiply(double k, Matrix m)	Mengalikan semua elemen dalam matriks m dengan skalar k.
Transpose(Matriks m)	Menghasilkan matriks transpose dengan menukar baris dan kolom dari matriks m

3.1.2.3. Accessors

Nama Metode	Garis Besar Cara Kerja
Get(int i , int j)	Mengambil nilai elemen pada baris i dan kolom j dari matriks.
Set(int i , int j , double v)	Mengubah nilai elemen pada baris i dan kolom j menjadi nilai v.
GetRow(int i)	Mengambil seluruh elemen pada baris i dalam bentuk array.
GetColumn(int j)	Mengambil seluruh elemen pada kolom j dalam bentuk array.
SetRow(int i , double[] rowData)	Mengatur nilai elemen di baris i dengan data dari array rowData.
SetColumn(int j , double[] columnData)	Mengatur nilai elemen di kolom j dengan data dari array columnData.
GetRowCount()	Mengembalikan jumlah baris dalam matriks.
GetColumnCount()	Mengembalikan jumlah kolom dalam matriks.
size()	Mengembalikan jumlah elemen total dalam matriks.

3.1.2.4. Constructors and Clone

Nama Metode	Garis Besar Cara Kerja
Matrix(int nRows, int nColumns)	Konstruktor yang membuat matriks dengan ukuran nRows x nColumns, dengan semua elemen diinisialisasi ke 0.
Matriks(double[][] data)	Konstruktor yang membuat matriks berdasarkan data dua dimensi yang diberikan.
clone()	Mengembalikan salinan (deep copy) dari matriks saat ini.

3.2 Class DeterminantCofactorSolver

3.2.1. Atribut

Nama Atribut	Penjelasan Singkat
n	Jumlah kolom dari matriks m, digunakan untuk menentukan ukuran matriks dan untuk menghitung determinan dalam proses.
det	Variabel yang menyimpan hasil akhir perhitungan determinan pada setiap tahap ekspansi kofaktor.
sign	Variabel yang menentukan apakah kofaktor akan ditambah atau dikurangkan dalam ekspansi kofaktor (bergantung pada posisinya).
cofactor	Nilai elemen pada baris dan kolom tertentu dari matriks, digunakan untuk ekspansi kofaktor.
minor	Submatriks yang didapat dengan menghilangkan satu baris dan satu kolom dari matriks utama, digunakan untuk menghitung determinan secara rekursif.

3.2.2. Metode

Nama Metode	Garis Besar Cara Kerja
determinant(Matrix m)	Menghitung determinan dari matriks m dengan menggunakan metode ekspansi kofaktor. Jika matriks berukuran 1x1, langsung mengembalikan nilainya. Jika matriks 2x2, menggunakan rumus determinan khusus untuk ukuran tersebut. Untuk ukuran lebih besar, dilakukan ekspansi kofaktor secara rekursif.
getMinor(Matrix m, int row, int column)	Menghasilkan submatriks (minor) dan matriks m dengan menghilangkan baris row dan kolom column. Submatriks ini digunakan untuk proses perhitungan determinan rekursif.

3.3 Class DeterminantOBESolver

3.3.1. Atribut

Nama Atribut	Penjelasan Singkat
matrix	Matriks yang akan dihitung determinannya menggunakan metode eliminasi baris elementer (OBE).
solution	Variabel yang menyimpan hasil akhir dari perhitungan determinan.
multiplier	Digunakan untuk melacak perubahan tanda dan skala saat mengubah matriks selama proses eliminasi baris elementer (OBE).
leadOneRow	Menunjukkan baris utama dalam b proses eliminasi untuk memastikan posisi elemen utama.
leadOneCol	Menunjukkan kolom utama dalam

	proses eliminasi untuk memastikan posisi elemen utama.
zeroCol	Flag yang menunjukkan apakah suatu kolom bernilai nol, yang digunakan dalam pengecekan selama proses eliminasi.

3.3.2. Metode

Nama Metode	Garis Besar Cara Kerja
Solve(Matrix m)	Metode statis untuk menghitung determinan dengan menciptakan objek DeterminantOBESolver dan memanggil metode instance Solve().
Solve()	Metode determinan matriks dengan menggunakan eliminasi baris elementer (OBE). Menyusun matriks menjadi bentuk segitiga atas dan menghitung hasilnya.
GetSolution()	Mengembalikan nilai dari determinan yang telah dihitung dan disimpan dalam atribut solution
DeterminantOBESolver(Matrix m)	Konstruktor yang menginisialisasi solver dengan matriks m dan mengatur solution menjadi nol sebagai nilai awal

3.4 Class BicubicSplineInterpolator

3.4.1. Atribut

Nama Atribut	Penjelasan Singkat
function	Matriks satu kolom hasil konversi dari matriks input yang berisi nilai-nilai fungsi yang akan digunakan dalam interpolasi.
equation	Matriks persamaan 16x16 yang dihasilkan dari evaluasi titik-titik

	fungsi dan turunannya untuk interpolasi bicubic.
coeffs	Array berisi koefisien hasil penyelesaian persamaan dengan metode Gauss-Jordan untuk interpolasi.
rgb	Array berisi nilai-nilai RGB yang diinterpolasi untuk gambar dalam metode bicubicInterpolateRGB
mValues	Matriks 4x4 yang berisi nilai fungsi dan turunannya di titik-titik grid untuk interpolasi bicubic warna.

3.4.2. Metode

Nama Metode	Garis Besar Cara Kerja
convert_matrix_16(Matrix m)	Mengubah matriks 4x4 menjadi matrix 16x1 dengan cara menggabungkan elemen-elemen dari baris-baris matriks secara berurutan.
count(double[] koef, double x, double y)	Menghitung nilai interpolasi pada titik (x,y) menggunakan koefisien dari persamaan bicubic.
bicubicInterpolate(Matrix m, double a, double b)	Melakukan interpolasi bicubic dengan mengonversi matriks m menjadi sistem persamaan, menyelesaikan dengan Gauss-Jordan dan menghitung hasil interpolasi pada (a, b).
bicubicInterpolateRGB(double[][][] pixels, int x0, int y0, double dx, double dy)	Menginterpolasi warna RGB dengan bicubic pada gambar, menghitung warna pada posisi relatif (dx, dy) dari titik (x0, y0).
getPixelValue(double[][][] pixels, int x, int y , int c)	Mengambil nilai pixel pada koordinat (x,y) untuk komponen warna c(merah, hijau, atau biru) dari matriks gambar pixels.
clamp(int val, int min , int max)	Membatasi nilai integer val dalam rentang [min,max].

clamp(double val, double min, double max)	Membatasi nilai double val dalam rentang [min,max].

3.5 Class ImageResizer

3.5.1. Atribut

Nama Atribut	Penjelasan Singkat
inputImagePath	Jalur lokasi file gambar input yang akan diubah ukurannya
outputImagePath	Jalur lokasi file gambar output yang akan disimpan setelah proses perubahan ukuran.
scaleWidth	Faktor skala untuk lebar gambar yang mengatur seberapa besar atau kecil gambar diubah ukurannya secara horizontal.
scaleHeight	Faktor skala untuk tinggi gambar yang mengatur seberapa besar atau kecil gambar diubah ukurannya secara vertikal.
inputImage	Objek BufferedImage yang memuat gambar asli dari file yang diambil berdasarkan jalur input.
originalWidth	Lebar asli dari gambar input sebelum diubah ukurannya.
originalHeight	Tinggi asli dari gambar input sebelum diubah ukurannya.
newWidth	Lebar baru dari gambar setelah proses perubahan ukuran, dihitung dari faktor skala..
newHeight	Tinggi baru dari gambar setelah proses perubahan ukuran, dihitung dari faktor skala.
outputImage	Objek BufferedImage yang memuat gambar hasil resize dengan dimensi baru.

pixelData	Matrix 3D yang menyimpan nilai RGB dari setiap pixel pada gambar input.
rgbValue	Nilai integer RGB dari pixel hasil interpolasi yang akan dituliskan pada gambar output.
r,g,b	Komponen warna merah, hijau , dan biru yang diambil dari gambar input atau hasil interpolasi untuk pixel tertentu.

3.5.2. Metode

Nama Metode	Garis Besar Cara Kerja
bicubicInterpolateRGB(double[][][] pixels, int x0, int y0, double dx, double dy)	Menginterpolasi warna RGB pada gambar input menggunakan metode bicubic untuk menghasilkan pixel yang diubah ukurannya.
ImageIO.read(File file)	Membaca gambar dari file input dan memuatnya ke dalam objek BufferedImage.
ImageIO.write(BufferedImage image, String formatName, File output)	Menyimpan gambar yang telah diubah ukurannya ke dalam file dengan format tertentu.

3.6 Class PolynomialInterpolator

3.6.1. Atribut

Nama Atribut	Penjelasan Singkat
n	Jumlah baris dalam matrix input m , yang merepresentasikan jumlah titik data yang digunakan untuk interpolasi.
A	Matriks Vandermonde berukuran nxn, digunakan untuk membangun sistem persamaan berdasarkan

	nilai-nilai x pada titik data.
b	Matriks n x 1 yang berisi nilai-nilai y dari titik data, digunakan sebagai matriks hasil dalam sistem persamaan.
xi	Nilai x dari titik data ke-i dalam matriks m.
yi	Nilai y dari titik data ke-i dalam matriks m.
augmentedMatrix	Matriks augmented yang menggabungkan matriks A dengan matriks b untuk diselesaikan menggunakan metode eliminasi Gauss.
coeffs	Array yang menyimpan solusi dari sistem persamaan, yaitu koefisien polinomial hasil interpolasi
y	Hasil nilai interpolasi pada titik x yang diberikan.

3.6.2. Metode

Nama Metode	Garis Besar Cara Kerja
interpolate(Matrix m, double x)	Melakukan interpolasi polinomial dengan membangun matriks Vandermonde dari data input, menyelesaikan sistem persamaan dengan Gauss, dan menghitung hasil interpolasi pada titik x.
Matrix.Append(Matrix A, Matrix b)	Menggabungkan matriks A dan b untuk membentuk matriks augmented yang digunakan dalam eliminasi Gauss.
GaussSolver.Solve()	Melakukan eliminasi Gauss pada matriks augmented untuk mendapatkan koefisien polinomial.
solver.GetSingleSolution()	Mengembalikan solusi tunggal dari sistem persamaan yang telah diselesaikan menggunakan eliminasi

	Gauss.

3.7 Class InverseNormalSolver

3.7.1. Atribut

Nama Atribut	Penjelasan Singkat
determinant	Nilai determinan dari matriks m , digunakan untuk memastikan bahwa matriks memiliki invers dan untuk menghitung invers matriks.
adjoint	Matriks yang menyimpan nilai kofaktor matriks asli, digunakan untuk menghitung invers melalui transposisi dan pembagian dengan determinan.

3.7.2. Metode

Nama Metode	Garis Besar Cara Kerja
Solve(Matrix m)	Menghitung invers dari matriks m dengan metode invers normal. Pertama, menghitung determinan untuk memastikan matriks memiliki invers, kemudian menghitung matriks adjoin (kofaktor dan transposisi) dan membaginya dengan determinan.
DeterminantCofactorSolver.determinant(Matrix m)	Menghitung determinan dari matriks m menggunakan metode ekspansi kofaktor, yang diperlukan untuk menentukan apakah invers ada dan menghitung matriks adjoint.
Matrix.GetMinor(int row, int col)	Menghasilkan minor dari matriks m dengan menghilangkan baris dan kolom tertentu, digunakan untuk menghitung nilai kofaktor pada adjoint.

Matrix.Transpose (Matrix m)	Transpose dari matriks adjoint, yang merupakan langkah penting dalam menghitung invers matriks.
Matrix.Multiply(double scalar, Matrix m)	Mengalikan matriks adjoint dengan kebalikan dari determinan untuk mendapatkan invers matriks.

3.8 Class InverseOBESolver

3.8.1. Atribut

Nama Atribut	Penjelasan Singkat
matrix	Matriks asli yang akan dihitung inversnya menggunakan metode Gauss-Jordan
siInvers	Matriks yang menyimpan hasil invers dari matriks asli setelah dipecahkan oleh metode Gauss-Jordan.
siMerge	Matriks augmented yang terdiri dari matriks asli yang digabungkan dengan matriks identitas, digunakan dalam proses Gauss-Jordan untuk menemukan invers.

3.8.2. Metode

Nama Metode	Garis Besar Cara Kerja
Solve(Matrix m)	Menginisialisasi objek InverseOBESolver dan memanggil metode Solve() untuk menghitung invers dari matriks m menggunakan metode eliminasi Gauss-Jordan
Solve()	Melakukan eliminasi Gauss-Jordan pada matriks augmented (gabungan matriks asli dan identitas) untuk menghitung matriks invers.

GetMatrix()	Mengembalikan matriks asli yang sedang diproses oleh solver..
Matrix.Append(Matrix a, Matrix b)	Menggabungkan matriks a dengan matriks b untuk membuat matriks augmented yang digunakan dalam proses Gauss-Jordan.
GaussJordanSolver.Solve()	Menyelesaikan matriks augmented menggunakan metode eliminasi Gauss-Jordan untuk menemukan solusi, dalam hal ini inver matriks.

3.9 Class CramerSolver

3.9.1. Atribut

Nama Atribut	Penjelasan Singkat
detM	Determinan dari matriks equationSpace yang merupakan matriks koefisien dari sistem persamaan linear.
result	Array yang menyimpan hasil solusi untuk setiap variabel dalam sistem persamaan setelah dihitung menggunakan aturan Cramer..

3.9.2. Metode

Nama Metode	Garis Besar Cara Kerja
GetSubmatrices(Matrix equationSpace, Matrix solutionSpace)	Metode ini menghasilkan submatriks yang diperlukan untuk aturan Cramer
Solve(Matrix equationSpace, Matrix solutionSpace)	Menghitung solusi dari sistem persamaan linear menggunakan aturan Cramer dengan cara menghitung determinan dan mengganti kolom matriks koefisien dengan kolom solusi.

sliceAndInsert(Matrix m1, Matrix m1, Matrix m2, int excludeCol)	Membuat matriks baru dengan mengganti kolom ke-excludeCol dari matriks m1 dengan kolom dari matriks m2. Digunakan untuk aturan Cramer dalam substitusi kolom.

3.10 Class GaussJordanSolver

3.10.1. Atribut

Nama Atribut	Penjelasan Singkat
detM	Determinan dari matriks equationSpace yang merupakan matriks koefisien dari sistem persamaan linear.
result	Array yang menyimpan hasil solusi untuk setiap variabel dalam sistem persamaan setelah dihitung menggunakan aturan Cramer..

3.10.2. Metode

Nama Metode	Garis Besar Cara Kerja
GetSubmatrices(Matrix equationSpace, Matrix solutionSpace)	Metode ini menghasilkan submatriks yang diperlukan untuk aturan Cramer
Solve(Matrix equationSpace, Matrix solutionSpace)	Menghitung solusi dari sistem persamaan linear menggunakan aturan Cramer dengan cara menghitung determinan dan mengganti kolom matriks koefisien dengan kolom solusi.
sliceAndInsert(Matrix m1, Matrix m1, Matrix m2, int excludeCol)	Membuat matriks baru dengan mengganti kolom ke-excludeCol dari matriks m1 dengan kolom dari matriks m2. Digunakan untuk aturan Cramer dalam substitusi kolom.

3.11 Class GaussSolver

3.11.1. Atribut

Nama Atribut	Penjelasan Singkat
matrix	Matriks augmented yang berisi sistem persamaan linear yang akan diselesaikan menggunakan metode eliminasi Gauss.

3.11.2. Metode

Nama Metode	Garis Besar Cara Kerja
GaussSolver(Matrix m)	Kontruktor yang membuat objek GaussSolver dan melakukan salinan dari matriks yang akan diselesaikan.
Solve(Matrix m)	Metode statis untuk memulai proses eliminasi Gauss pada matriks augmented m.
Solve()	Melakukan eliminasi Gauss pada matriks augmented untuk menyelesaikan sistem persamaan linear, mengubah matriks menjadi bentuk eselon.
GetSingleSolution()	Mengambil solusi tunggal dari sistem persamaan, jika ada, dengan mengembalikan array berisi nilai variabel solusi.
GetMultipleSolution()	Mengambil solusi multipel dari sistem persamaan, jika ada, dan mengembalikan array dua dimensi berisi nilai-nilai solusi.
GetMatrix()	Mengembalikan matriks augmented saat ini yang sedang diproses oleh solver.

3.12 Class InverseSolver

3.12.1. Atribut

Nama Atribut	Penjelasan Singkat
temp	Matriks sementara berukuran nx1 yang dibuat dari array b, mewakili sisi kanan sistem persamaan linear.
sum	Matriks hasil perkalian antara invers matriks m dan matriks temp, yang berisi solusi sistem persamaan linear.

3.12.2. Metode

Nama Metode	Garis Besar Cara Kerja
Solve(Matrix m, double[] b)	Menghitung solusi sistem persamaan linear dengan cara mengalikan invers dari matriks m dengan vektor b. Menggunakan InverseOBESolver untuk menghitung invers matriks.

3.13 Class OBEAdder

3.13.1. Atribut

Nama Atribut	Penjelasan Singkat
newMatrix	Matriks baru hasil operasi yang menyimpan perubahan setelah penambahan baris yang telah diskalakan dari baris lain.
newRowData	Array yang menyimpan data hasil operasi penambahan baris sourceRow yang telah diskalakan ke dalam baris targetRow.

3.13.2. Metode

Nama Metode	Garis Besar Cara Kerja
Add(Matrix m, int targetRow, int sourceRow, double scale)	Menambahkan baris sourceRow yang telah diskalakan dengan faktor scale ke baris targetRow dalam matriks m. Mengembalikan matriks baru yang telah dimodifikasi.

3.14 Class OBEScaler

3.14.1. Atribut

Nama Atribut	Penjelasan Singkat
scaledMatrix	Matriks hasil setelah baris yang ditentukan diskalakan dengan faktor scale, disalin dari matriks aslli.

3.14.2. Metode

Nama Metode	Garis Besar Cara Kerja
Scale(Matrix m, int row, double scale)	Menskalakan baris row dari matriks m dengan faktor scale, mengalikan setiap elemen dalam baris tersebut, dan mengembalikan matriks baru yang telah dimodifikasi.

3.15 Class OBESwitcher

3.15.1. Atribut

Nama Atribut	Penjelasan Singkat

cola	Matriks hasil salinan dari matriks m, digunakan untuk menukar dua baris tanpa mengubah matriks asli.
temp	Array yang menyimpan sementara baris rowA saat proses penukaran baris.

3.15.2. Metode

Nama Metode	Garis Besar Cara Kerja
Switch(Matrix m , int rowA, int rowB)	Menukar dua baris, rowA dan rowB, dalam matriks m dengan membuat salinan matriks, kemudian menukar posisi baris tersebut, dan mengembalikan matriks hasil.

3.16 Class LinearRegressor

3.16.1. Atribut

Nama Atribut	Penjelasan Singkat
matrix1s	Array 2D berisi data input untuk regresi linear, di mana kolom terakhir adalah nilai y dan kolom - kolom sebelumnya adalah nilai x.
matrixYx	Matriks kolom berisi nilai-nilai y dari data input, digunakan untuk membentuk sistem persamaan regresi.
X	Matriks X hasil dari penggabungan matrix1s dan kolom-kolom nilai x dari data input tanpa kolom y.
XT	Transposisi dari matriks X, digunakan untuk membentuk matriks augmented dalam sistem persamaan.
augmented	Matriks augmented yang berisi hasil

	perkalian XT * X dan XT * Y, digunakan untuk menyelesaikan persamaan regresi

3.16.2. Metode

Nama Metode	Garis Besar Cara Kerja
Solve(double[][] input)	Melakukan regresi linear dengan menggunakan eliminasi Gauss-Jordan untuk menyelesaikan sistem persamaan, mengembalikan koefisien hasil regresi.
deleteLastColumn(Matrix m)	Menghapus kolom terakhir dari matriks input, digunakan untuk membentuk matriks X dari data input tanpa nilai y.

3.17 Class QuadraticRegressor

3.17.1. Atribut

Nama Atribut	Penjelasan Singkat
matrix1s	Matriks dengan kolom yang isinya nilai 1, digunakan sebagai bagian dari matriks X dalam persamaan regresi kuadratik.
matrixYx	Matriks kolom berisi nilai-nilai y dari data input, digunakan untuk membentuk sistem persamaan regresi.
LinearTerms	Matriks yang berisi nilai-nilai linear dari variabel independen, digunakan untuk membentuk matriks X dalam regresi.
quadraticTerms	Matriks yang berisi nilai kuadrat dari variabel independen, digunakan dalam model regresi kuadratik.

interactionTerms	Matriks yang berisi hasil perkalian antara dua variabel independen yang berbeda, digunakan untuk menangkap interaksi antar variabel dalam model.
X	Matriks X hasil gabungan dari matrix1s, linearTerms, quadraticTerms, dan interactionTerms..
XT	Transposisi dari matriks X, digunakan untuk membentuk matriks augmented dalam sistem persamaan.
augmented	Matriks augmented yang berisi hasil perkalian XT * X dan XT * Y, digunakan untuk menyelesaikan persamaan regresi kuadratik.

3.17.2. Metode

Nama Metode	Garis Besar Cara Kerja
solve(double[][] input)	Melakukan regresi kuadratik dengan menggunakan eliminasi Gauss untuk menyelesaikan sistem persamaan, mengembalikan koefisien hasil regresi.
deleteLastColumn(Matrix m)	Menghapus kolom terakhir dari matriks input, digunakan untuk membentuk matriks X dari data input tanpa nilai y.
createQuadraticTerms(Matrix linearterms)	Membuat matriks yang berisi nilai kuadrat dari setiap variabel independen (linear terms).
createInteractionTerms(Matrix linearTerms)	Membuat matriks yang berisi interaksi antar variabel independen (perkalian antar dua variabel yang berbeda) dalam model regresi kuadratik.

BAB IV

EKSPERIMEN

4.1 Studi Kasus #1 (Temukan solusi SPL, Ax = b)

4.1.1

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

Output:

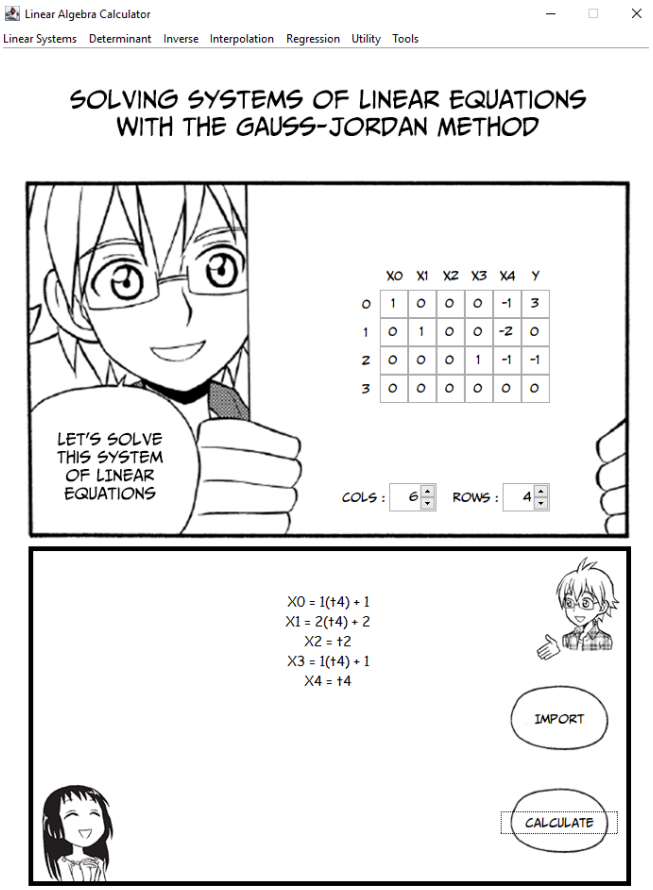
Dari gambar tersebut, matriks A dan vektor b membentuk sebuah sistem persamaan linear yang diselesaikan menggunakan eliminasi Gauss. Berdasarkan output yang ditunjukkan, sistem persamaan ini tidak memiliki solusi. Hal ini terjadi karena setelah melalui proses eliminasi, baris terakhir pada matriks augmented memiliki semua elemen nol kecuali kolom

hasilnya (kontradiksi), yang mengindikasikan bahwa sistem ini *inconsistent* atau bertentangan, sehingga tidak ada solusi yang memenuhi semua persamaan.

4.1.2

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

Output:



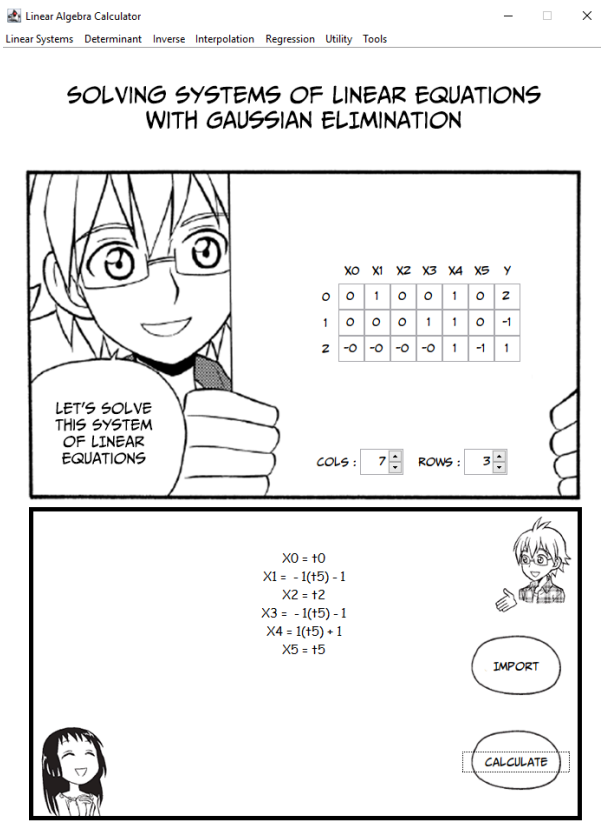
Pada kasus ini, sistem persamaan linear diselesaikan menggunakan metode Gauss-Jordan. Dari output yang diberikan, terlihat bahwa sistem memiliki solusi dengan variabel bebas, yaitu t_4. Hal ini menunjukkan bahwa sistem memiliki tak hingga banyak solusi karena adanya variabel bebas. Proses eliminasi menghasilkan bentuk matriks augmented yang

menunjukkan adanya hubungan linear antara variabel-variabel lain dengan variabel bebas t_4, sehingga solusi dapat diekspresikan dalam bentuk parametrik

4.1.3

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

Output:



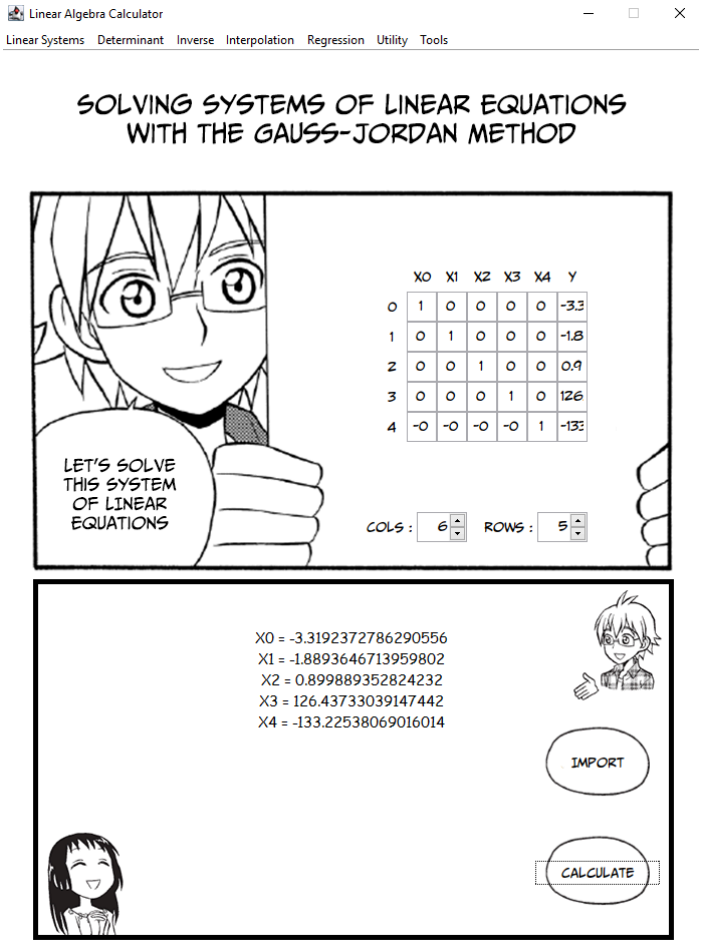
Pada kasus ini, sistem persamaan linear diselesaikan dengan eliminasi Gauss. Dari hasil output, terlihat bahwa sistem memiliki lebih dari satu variabel bebas, yaitu t_0 dan t_5 .Hal ini menunjukkan bahwa sistem memiliki tak hingga banyak solusi karena dua variabel bebas tersebut. Proses eliminasi menghasilkan hubungan linear antara variabel-variabel lain dengan $t_0$dan t_5 yang memungkinkan solusi diungkapkan dalam bentuk parametrik, dimana variabel lainnya dinyatakan sebagai fungsi dari variabel bebas.

4.1.4

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

Output:

Untuk n = 5:



Pada kasus ini, matriks H merupakan matriks Hilbert, yang dikenal dengan sifatnya yang kondisional buruk, terutama untuk nilai nnn yang besar. Sistem persamaan linear yang

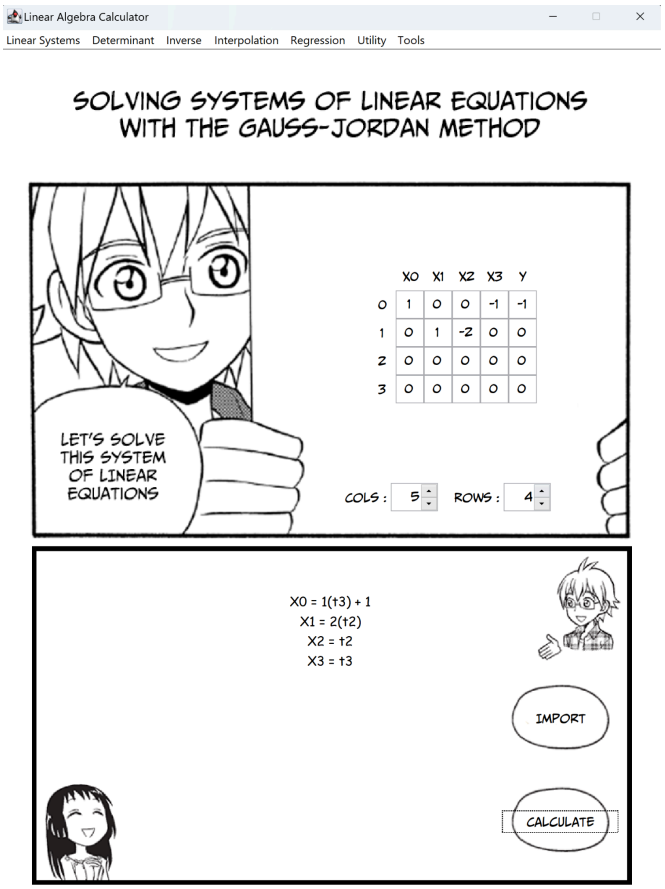
dibentuk oleh matriks Hilbert diselesaikan menggunakan metode Gauss-Jordan. Berdasarkan output untuk n = 5, hasil yang diperoleh menunjukkan nilai solusi yang sangat besar dan kecil secara bergantian, mengindikasikan bahwa perhitungan menjadi tidak stabil karena kondisi buruk matriks Hilbert tersebut. Hal ini umum terjadi pada sistem dengan matriks Hilbert karena sensitivitasnya terhadap kesalahan pembulatan.

4.2 Studi Kasus #2 (SPL Matriks Augmented)

4.2.1

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

Output:

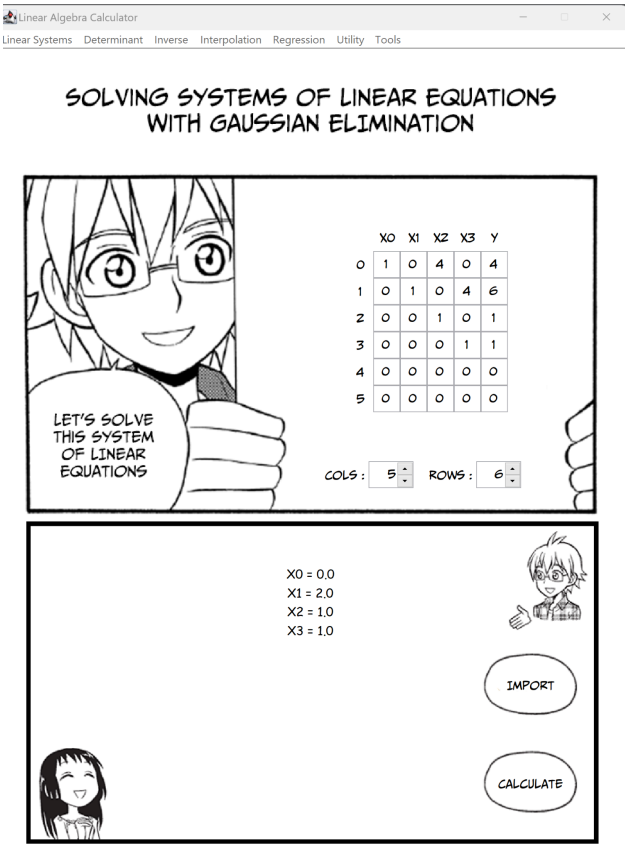


Pada kasus ini, sistem persamaan linear diselesaikan menggunakan metode Gauss-Jordan. Dari bentuk matriks augmented dan output yang diberikan, terlihat bahwa sistem ini memiliki variabel bebas, yaitu t_2 dan t_3. Sistem menghasilkan solusi tak hingga banyak karena adanya variabel bebas tersebut. Nilai variabel lainnya dinyatakan dalam fungsi dari variabel bebas t_2 dan t_3, di mana solusi akhirnya diberikan dalam bentuk parametrik.

4.2.2

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

Output:



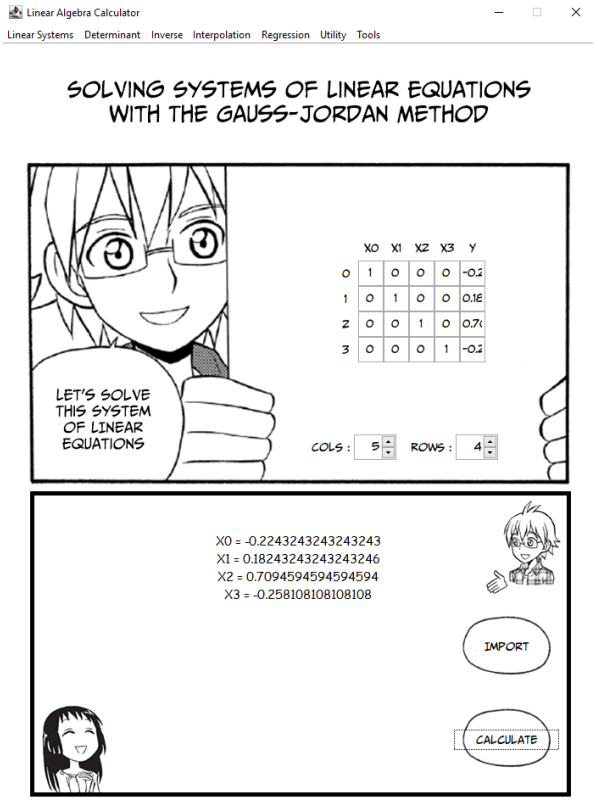
Pada kasus ini, sistem persamaan linear diselesaikan menggunakan metode eliminasi Gauss. Dari bentuk matriks augmented yang diberikan, metode ini bekerja dengan mengubah matriks menjadi bentuk segitiga atas (upper triangular), kemudian dilakukan substitusi balik (*back substitution*) untuk mendapatkan solusi dari setiap variabel. Berdasarkan output yang terlihat, sistem ini memiliki solusi unik dengan hasil yang spesifik untuk setiap variabel X_0, X_1,X_2,X_3. Solusi ini menunjukkan bahwa sistem tidak memiliki variabel bebas dan solusi dapat diperoleh secara langsung setelah proses eliminasi.

4.3 Studi Kasus #3 (SPL berbentuk)

4.3.1

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

Output:

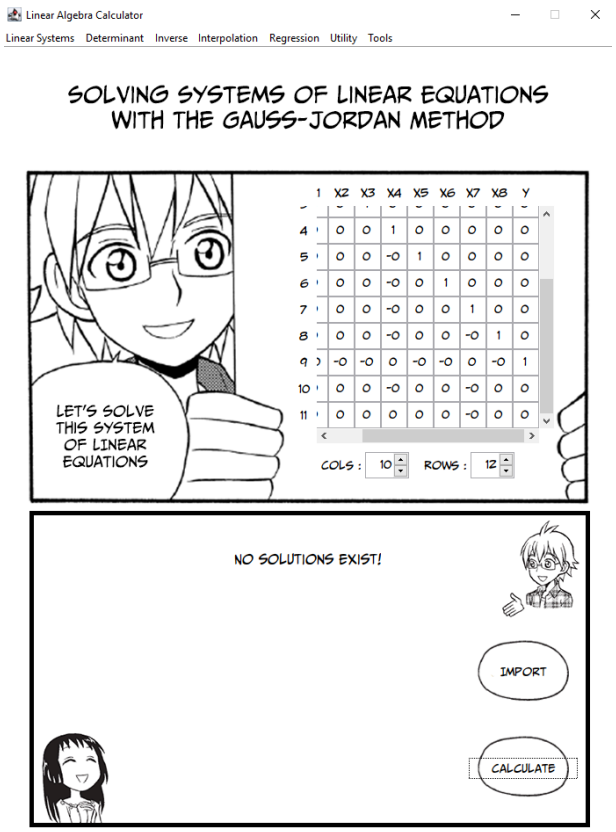


Pada kasus ini, sistem persamaan linear diselesaikan menggunakan metode Gauss-Jordan. Berdasarkan output yang diberikan, hasil perhitungan menghasilkan solusi yang spesifik untuk setiap variabel X_0, X_1, X_2, X_3 Tidak ada variabel bebas dalam sistem ini, yang berarti sistem memiliki solusi tunggal. Proses Gauss-Jordan berhasil mengubah matriks augmented ke bentuk eselon tereduksi, yang memungkinkan solusi dapat dihitung secara langsung dan tepat untuk setiap variabel.

4.3.2 Who made this and why do you hate us

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

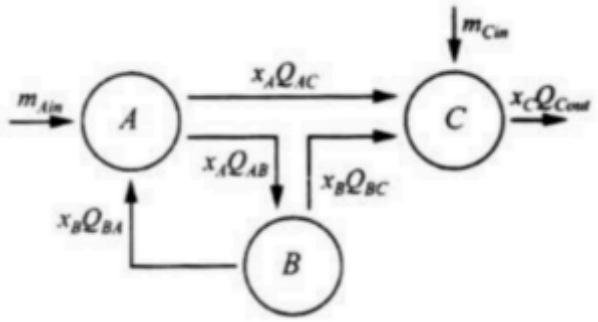
Output:



Pada kasus ini, sistem persamaan linear yang kompleks diselesaikan menggunakan metode Gauss-Jordan. Berdasarkan output yang diberikan, sistem persamaan ini ternyata tidak memiliki solusi. Hal ini terjadi karena, setelah melalui proses eliminasi Gauss-Jordan, ditemukan bahwa sistem ini **inconsistent** atau bertentangan, yang mengindikasikan bahwa

tidak ada nilai dari variabel-variabel yang dapat memenuhi semua persamaan secara bersamaan. Sistem yang tidak konsisten seringkali terjadi jika ada persamaan yang saling bertolak belakang atau redundansi yang menyebabkan konflik di antara persamaan.

4.4 Studi Kasus #4 (Sistem Reaktor)



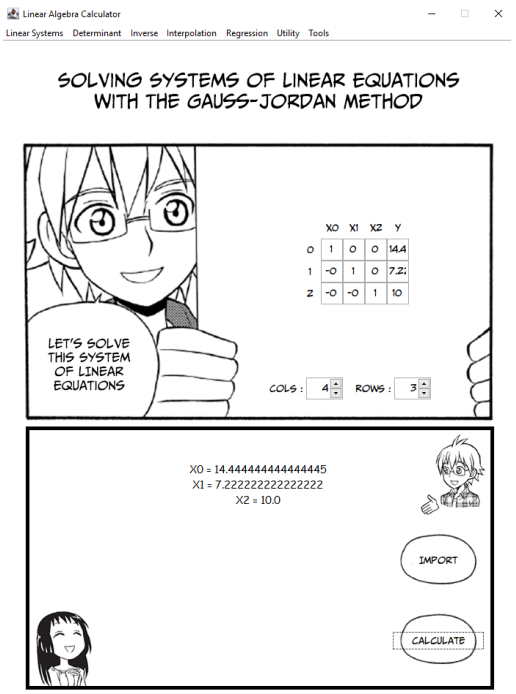
Dengan laju volume Q dalam m^3 /s dan input massa min dalam mg/s. Konservasi massa pada tiap inti reaktor adalah sebagai berikut:

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

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

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

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.



Pada sistem persamaan ini, solusi ditemukan dengan pertama memuat informasi ke dalam bentuk matriks augmented dimana kolom pertama berisi koefisien Xa, kolom kedua berisi koefisien Xb, kolom ketiga berisi koefisien Xc, dan kolom ketiga berisi hasil dari persamaannya. Dari penyelesaian matriks tersebut menggunakan metode Gauss-Jordan, didapatkan hasil Xa = 14,444…. Xb = 7.222…. dan Xc = 10.

4.5 Studi Kasus #5 (Interpolasi)

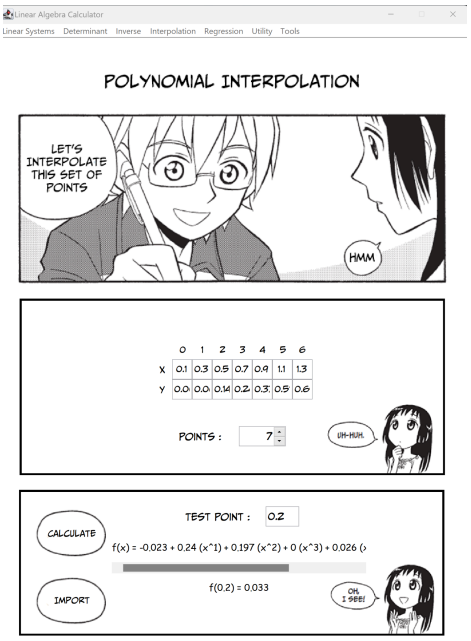
4.5.1 Titik diincar

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	0.37	0.518	0.697

Lakukan pengujian pada nilai-nilai berikut:

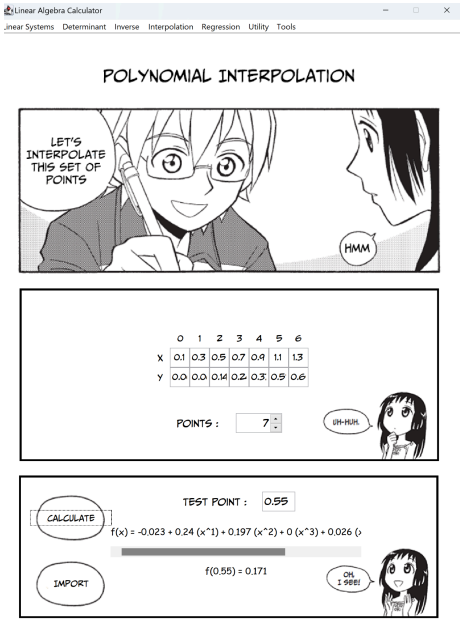
a. $x = 0.2 \quad f(x) = 0.033$



Pada kasus ini, dilakukan interpolasi polinomial berdasarkan tabel nilai x dan $f(x)$ yang diberikan. Tujuan dari interpolasi ini adalah untuk memperkirakan nilai fungsi $f(x)$ pada

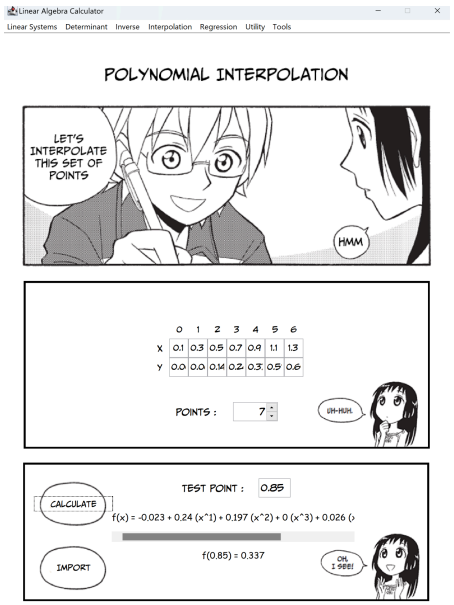
titik-titik yang tidak ada dalam tabel, seperti $x = 0.2$, yang menghasilkan $f(x) = 0.033$ berdasarkan hasil perhitungan program.

b. $x = 0.55 \quad f(x) = 0.171$



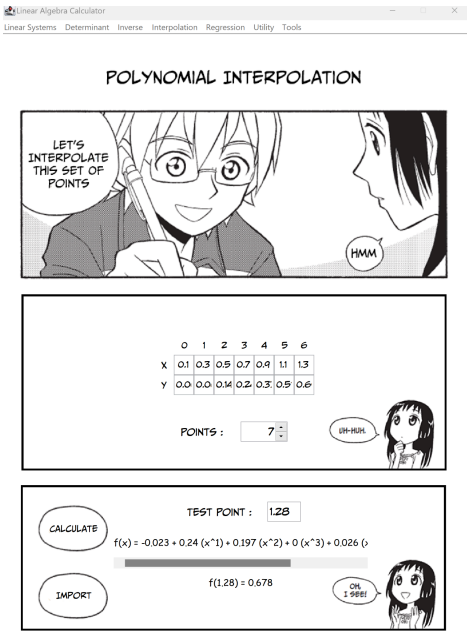
Pada kasus ini, dilakukan interpolasi polinomial berdasarkan tabel nilai x dan $f(x)$ yang diberikan. Tujuan dari interpolasi ini adalah untuk memperkirakan nilai fungsi $f(x)$ pada titik-titik yang tidak ada dalam tabel, seperti $x = 0.55$, yang menghasilkan $f(x) = 0.171$ berdasarkan hasil perhitungan program.

c. $x = 0.85 \quad f(x) = 0.337$



Pada kasus ini, dilakukan interpolasi polinomial berdasarkan tabel nilai x dan $f(x)$ yang diberikan. Tujuan dari interpolasi ini adalah untuk memperkirakan nilai fungsi $f(x)$ pada titik-titik yang tidak ada dalam tabel, seperti $x = 0.85$, yang menghasilkan $f(x) = 0.0337$ berdasarkan hasil perhitungan program.

d. $x = 1.28$ $f(x) = 0.678$



Pada kasus ini, dilakukan interpolasi polinomial berdasarkan tabel nilai x dan $f(x)$ yang diberikan. Tujuan dari interpolasi ini adalah untuk memperkirakan nilai fungsi $f(x)$ pada titik-titik yang tidak ada dalam tabel, seperti $x = 1.28$, yang menghasilkan $f(x) = 0.678$ berdasarkan hasil perhitungan program.

4.5.2 Kasus COVID-19

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:

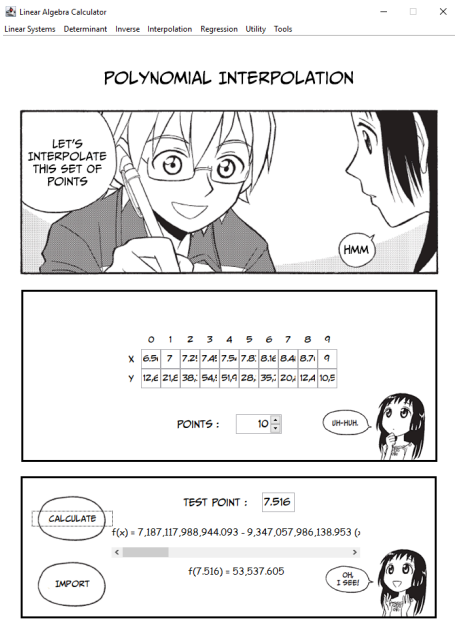
Tanggal (desimal) = bulan + (tanggal / jumlah hari pada bulan tersebut)

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:

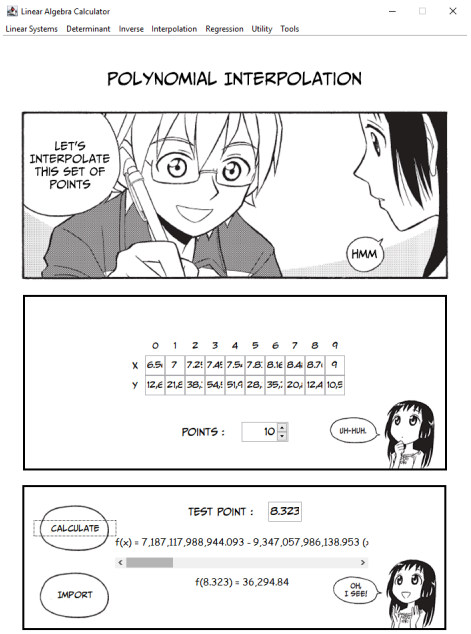
a. 16/07/2022



Pada kasus ini, dilakukan interpolasi polinomial untuk memprediksi jumlah kasus Covid-19 pada tanggal tertentu yang tidak terdapat dalam tabel data. Menggunakan tanggal yang sudah diubah ke bentuk desimal, proses interpolasi ini bertujuan untuk memperkirakan jumlah kasus pada tanggal 16 Juli 2022.

Berdasarkan gambar yang ditampilkan, nilai prediksi yang diperoleh melalui interpolasi polinomial untuk tanggal desimal $x = 7.516$ memberikan hasil sejumlah 53,537.605 kasus baru Covid-19. Proses interpolasi ini memungkinkan kita untuk memanfaatkan data yang ada untuk melakukan prediksi pada tanggal-tanggal lain yang tidak disertakan secara langsung dalam tabel asli, dengan asumsi bahwa tren data mengikuti pola yang dapat diwakili oleh polinom interpolasi.

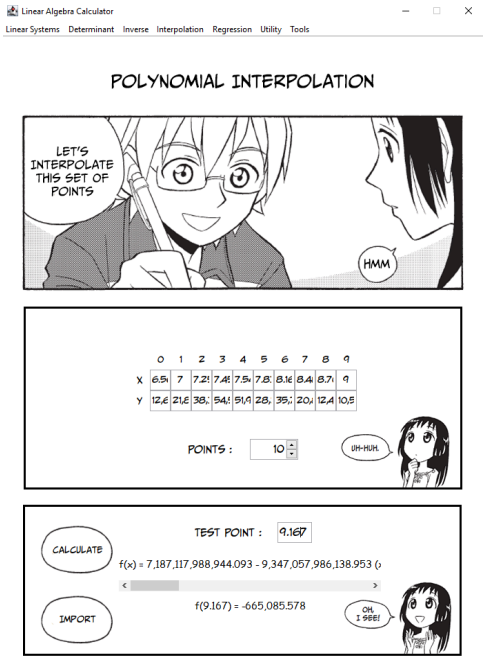
b. 10/08/2022



Pada kasus ini, dilakukan interpolasi polinomial untuk memprediksi jumlah kasus Covid-19 pada tanggal tertentu yang tidak terdapat dalam tabel data. Menggunakan tanggal yang sudah diubah ke bentuk desimal, proses interpolasi ini bertujuan untuk memperkirakan jumlah kasus pada tanggal 10 Agustus 2022.

Berdasarkan gambar yang ditampilkan, nilai prediksi yang diperoleh melalui interpolasi polinomial untuk tanggal desimal $x = 8.323$ memberikan hasil sejumlah 36,294.84 kasus baru Covid-19. Proses interpolasi ini memungkinkan kita untuk memanfaatkan data yang ada untuk melakukan prediksi pada tanggal-tanggal lain yang tidak disertakan secara langsung dalam tabel asli, dengan asumsi bahwa tren data mengikuti pola yang dapat diwakili oleh polinom interpolasi.

c. 05/09/2022



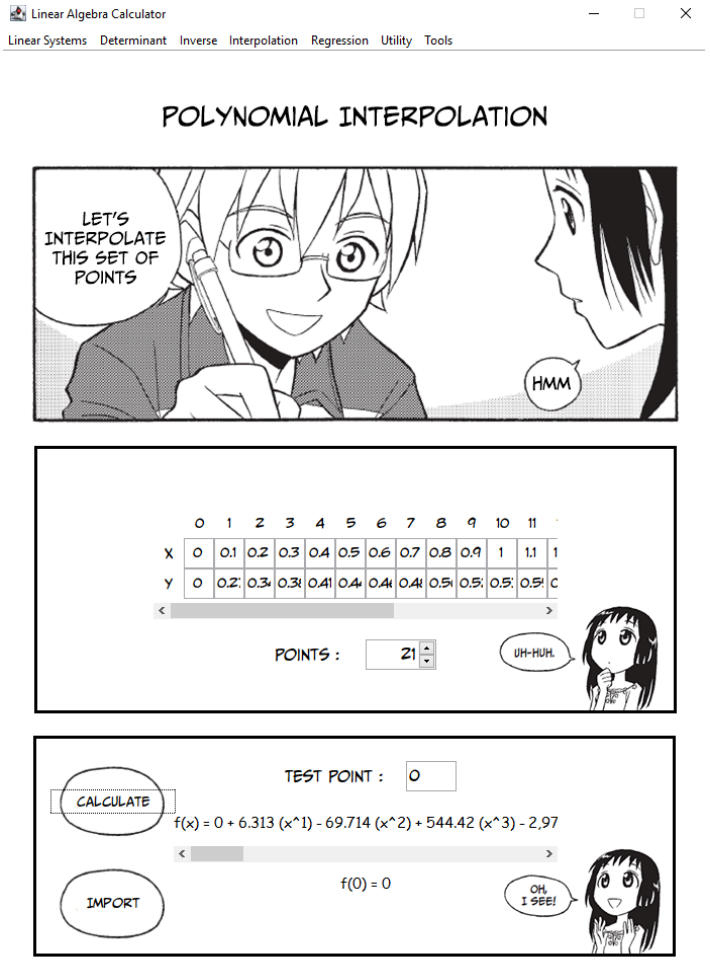
Pada kasus ini, angka mencapai negatif karena jarak antara titik data terlalu dekat. Interpolasi polinomial mencoba untuk melewati setiap titik dengan menyesuaikan derajat polinomial dengan jumlah titik yang disediakan. Akibatnya, jika terdapat banyak titik yang terletak pada posisi yang dekat, fungsi polinomial akan berosilasi dengan sangat cepat untuk dapat melewati semua titik dengan derajat polinomial yang tinggi.

4.5.3 Penyederhanaan $f(x)$

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.



Menggunakan interval 0.1 sehingga membentuk n = 20 memberikan 20 titik unik yang dapat diinterpolasi secara polinomial. Hasil interpolasi polinomial untuk 20 titik di kurva fungsi f(x) yang memiliki interval 0.1 pada selang [0, 2] memberikan hasil persamaan berikut.

f(x) = 0 + 6.313 (x^1) - 69.714 (x^2) + 544.42 (x^3) - 2,972.506 (x^4) + 11,732.645 (x^5) - 34,425.66 (x^6) + 76,549.102 (x^7) - 130,384.8 (x^8) + 170,400.759 (x^9) - 169,084.704 (x^10) + 123,197.325 (x^11) - 59,696.528 (x^12) + 11,496.559 (x^13) + 8,730.466 (x^14) - 9,684.825 (x^15) + 4,972.097 (x^16) - 1,603.934 (x^17) + 331.868 (x^18) - 40.591 (x^19) + 2.245 (x^20)

4.6 Studi Kasus #6 (Linear Kuadratik Berganda)

Diberikan sekumpulan data sesuai pada tabel berikut ini.

Table 12.1: Data for Example 12.1

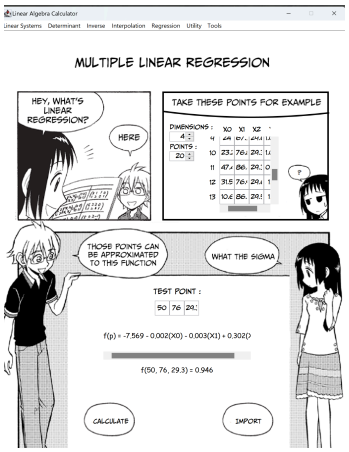
Nitrous Oxide, y	Humidity, x_1	Temp., x_2	Pressure, x_3	Nitrous Oxide, y	Humidity, x_1	Temp., x_2	Pressure, x_3
0.90	72.4	76.3	29.18	1.07	23.2	76.8	29.38
0.91	41.6	70.3	29.35	0.94	47.4	86.6	29.35
0.96	34.3	77.1	29.24	1.10	31.5	76.9	29.63
0.89	35.1	68.0	29.27	1.10	10.6	86.3	29.56
1.00	10.7	79.0	29.78	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.

Quadratic Regression : 0.944

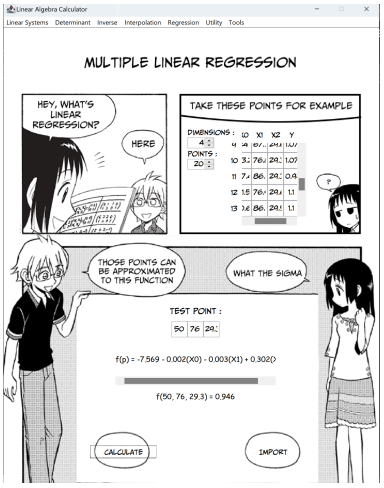


Pada contoh ini, dilakukan regresi linier berganda berdasarkan data yang terdapat pada tabel untuk memperkirakan nilai variabel dependen (Nitrous Oxide) berdasarkan variabel-variabel independen (Humidity, Temperature, Pressure). Dengan menggunakan Normal Estimation Equation for Multiple Linear Regression, diperoleh sistem persamaan linier yang sesuai dengan data.

Hasil regresi tersebut memberikan dua jenis regresi, yaitu Quadratic Regression dengan nilai R^2 = 0.944 , dan Linear Regression dengan nilai R^2 = 0.946. Nilai R^2 ini menunjukkan

seberapa baik model regresi dapat menjelaskan variasi dari data yang diberikan. Semakin mendekati 1, semakin baik model tersebut. Berdasarkan hasil ini, kedua model menunjukkan tingkat kesesuaian yang tinggi terhadap data yang diberikan.

Linear Regression : 0.946

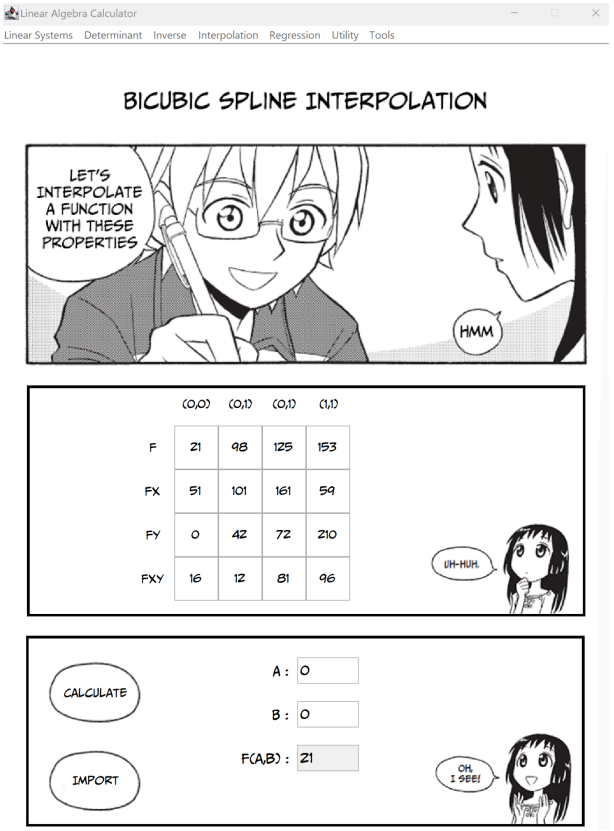


4.7 Studi Kasus #7 (Interpolasi Bicubic Spline)

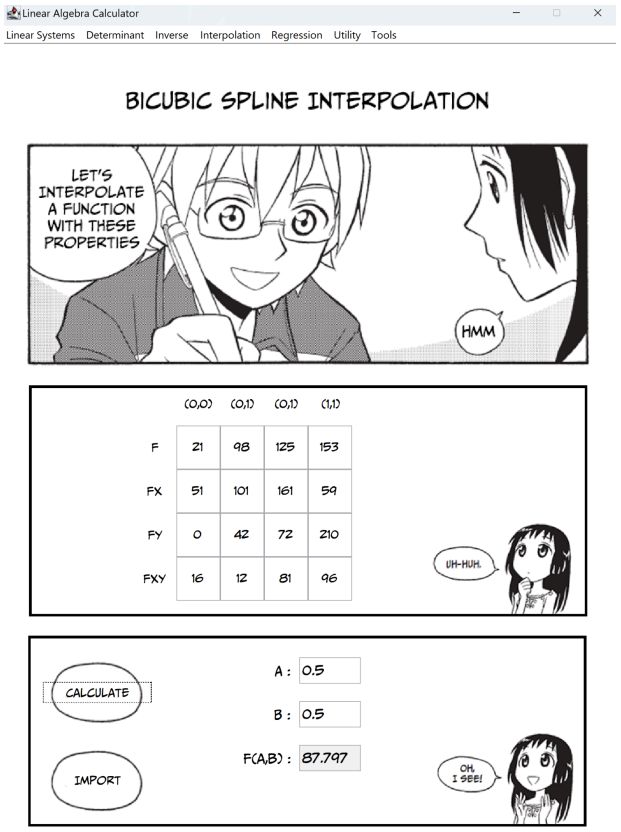
Diberikan matriks input dengan bentuk sebagai berikut. Format matriks masukan bukan mewakili nilai matriks, tetapi mengikuti format masukan pada bagian "Spesifikasi Tugas" nomor 7.

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

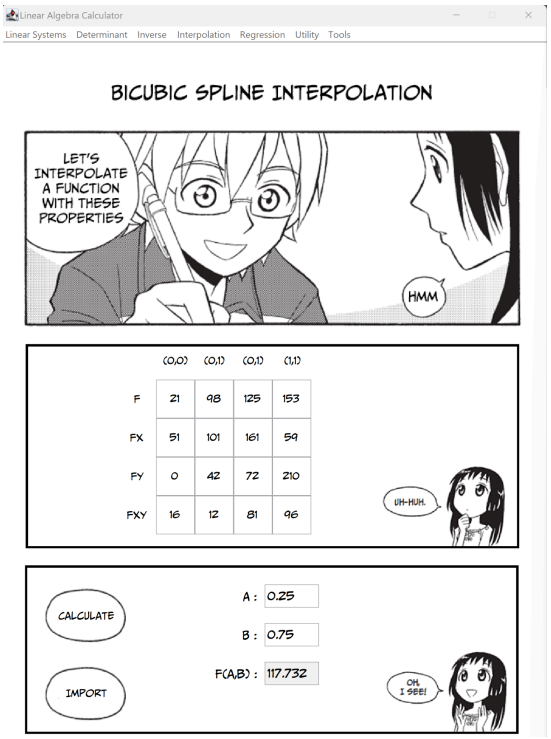
a. $f(0,0)$ = 21



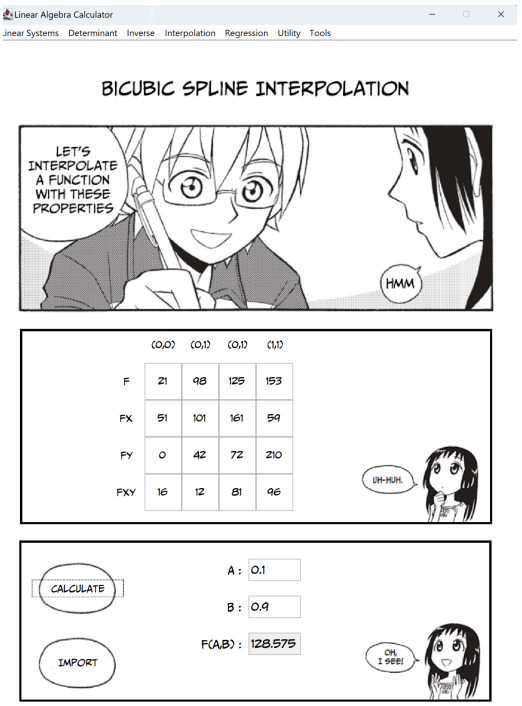
b. $f(0.5, 0.5) = 87.797$



c. $f(0.25, 0.75)$ = 117.732



d. $f(0.1, 0.9)$ = 128.575



Analisis dari studi kasus ini menunjukkan bahwa matriks input berfungsi sebagai representasi nilai-nilai pada titik-titik tertentu di bidang 2D. Metode interpolasi bicubic digunakan untuk menghitung nilai di antara titik-titik tersebut dengan lebih halus, berdasarkan derivatif dan nilai fungsi di sekitar titik. Matriks ini tidak menunjukkan nilai sebenarnya, melainkan

digunakan untuk memfasilitasi perhitungan interpolasi dengan menggunakan metode yang dijelaskan pada spesifikasi tugas.

BAB V

KESIMPULAN

5.1 Kesimpulan

Dalam tugas besar ini, kami berhasil mengembangkan program aljabar linear yang mampu menyelesaikan berbagai permasalahan matematis, termasuk:

- **Sistem Persamaan Linear**: Menyediakan solusi dengan metode Eliminasi Gauss, Gauss-Jordan, Cramer, dan Invers.
- **Determinan**: Menghitung nilai determinan menggunakan metode Kofaktor dan OBE.
- **Matriks Invers**: Menghitung invers matriks dengan metode Operasi Baris Elementer dan Metode Normal.
- **Interpolasi**: Melakukan interpolasi polinom dan bicubic spline.
- **Regresi**: Melakukan regresi linear berganda dan regresi kuadratik berganda.

Implementasi algoritma yang efisien dan penggunaan antarmuka GUI membuat program ini user-friendly dan dapat membantu pengguna dalam memahami konsep aljabar linear. Pengujian program menunjukkan bahwa hasil yang diperoleh akurat dan konsisten dengan perhitungan manual.

5.2 Komentar

Selama proses pengembangan program ini, kami menemukan bahwa mengintegrasikan konsep matematika dengan pemrograman memerlukan perhatian khusus terhadap detail. Beberapa komentar yang dapat kami sampaikan adalah:

- **Kompleksitas Algoritma**: Beberapa metode yang diimplementasikan, seperti metode Cramer dan invers matriks untuk ukuran besar, memiliki kompleksitas komputasi yang tinggi. Hal ini menyebabkan program menjadi kurang efisien pada matriks berukuran besar.
- **Keterbatasan Metode**: Metode Cramer dan invers tidak selalu praktis untuk sistem persamaan linear dengan ordo tinggi. Oleh karena itu, penting untuk memahami kapan sebaiknya menggunakan metode tertentu.

- **Validasi Input**: Pentingnya validasi input pengguna agar program dapat menangani kasus-kasus seperti matriks singular atau sistem yang tidak memiliki solusi unik.
- **Antarmuka Pengguna**: Meskipun GUI membantu dalam interaksi dengan pengguna, desain antarmuka yang lebih intuitif dan responsif akan meningkatkan pengalaman pengguna secara keseluruhan.

5.3 Refleksi

Proyek ini memberikan kami kesempatan untuk menerapkan teori aljabar linear dalam konteks praktis. Beberapa refleksi yang kami peroleh adalah:

- **Penguasaan Materi**: Implementasi algoritma aljabar linear membantu kami memperdalam pemahaman terhadap materi kuliah. Melalui coding, kami lebih memahami bagaimana algoritma bekerja secara detail.
- **Peningkatan Keterampilan Pemrograman**: Menggunakan Java dan library Swing untuk GUI meningkatkan keterampilan kami dalam pemrograman berorientasi objek dan pengembangan antarmuka pengguna.
- **Kerja Tim**: Proyek ini menekankan pentingnya komunikasi dan kolaborasi dalam tim. Pembagian tugas yang efektif dan saling membantu saat menghadapi kesulitan teknis sangat berkontribusi terhadap keberhasilan proyek.
- **Manajemen Waktu**: Kami belajar untuk mengatur waktu dengan lebih baik, mengingat banyaknya fitur yang harus diimplementasikan dalam waktu yang terbatas.
- **Pemecahan Masalah**: Tantangan yang dihadapi selama pengembangan, seperti debugging dan optimasi algoritma, mengasah kemampuan kami dalam pemecahan masalah dan berpikir kritis.

5.4 Saran

Untuk pengembangan lebih lanjut, beberapa saran yang dapat dipertimbangkan:

- **Optimasi Algoritma:** Implementasi algoritma yang lebih efisien untuk operasi matriks besar, seperti penggunaan pustaka matematika yang dioptimalkan.
- **Ekstensi Fitur:** Menambahkan fitur-fitur lanjutan seperti analisis eigenvalue dan eigenvector.
- **Validasi Input:** Meningkatkan mekanisme validasi input untuk memastikan keandalan program.

- **Dokumentasi:** Menyediakan dokumentasi yang lebih lengkap dan contoh penggunaan untuk memudahkan pengguna baru.
- **Pengembangan Lebih Lanjut**: Menambahkan fitur untuk metode numerik lainnya, seperti metode iteratif untuk sistem persamaan linear.
- **Optimasi Lebih Lanjut**: Melakukan optimasi lebih lanjut pada algoritma untuk meningkatkan efisiensi pada data berukuran sangat besar.
- **Dokumentasi Lebih Detail**: Menyediakan dokumentasi pengguna yang lebih detail, termasuk panduan penggunaan dan interpretasi hasil.

LAMPIRAN

Video for Bonus : Program Goes On (cringe warning, I can't sing) | A Porter Robi…

Github Repository : Linear Algebra Calculator feat. Anime Girls

Image Resizer

227 x 440 Pixels

440 x 440 pixels

source: Furina | Wikipedia