Laporan Tugas Besar 1

IF2123 Aljabar Linier dan Geometri

Sistem Persamaan Linier, Determinan, dan Penerapannya

Semester I Tahun 2025/2026

v1.0

DEADLINE: JUMAT, 2025/10/10 23:59 WIB

Laboratorium Ilmu dan Rekayasa Komputasi

Program Studi Teknik Informatika

Sekolah Teknik Elektro dan Informatika

Institut Teknologi Bandung

1 Pendahuluan

1.1 Rumusan Masalah

Aplikasi seperti Photomath telah menjadi solusi populer dalam membantu menyelesaikan berbagai permasalahan matematika, termasuk soal matriks dan sistem persamaan linear yang kompleks. Kemampuan aplikasi tersebut dalam memberikan langkah-langkah penyelesaian secara otomatis tidak terlepas dari penerapan algoritma dan konsep matematika yang kuat, khususnya aljabar linear.

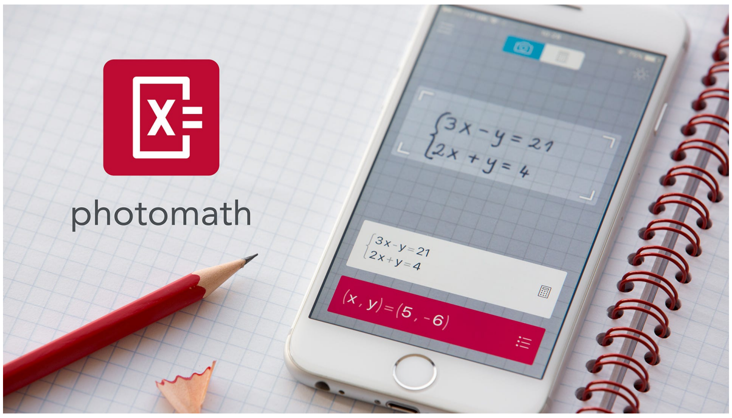

Figure 1: Ilustrasi aplikasi Photomath

Aljabar linear merupakan salah satu cabang matematika yang sangat penting dalam pengembangan ilmu komputer, teknik, fisika, ekonomi, dan berbagai bidang lainnya. Dalam kehidupan sehari-hari, konsep matriks dan sistem persamaan linear sering kali diaplikasikan untuk menyelesaikan berbagai persoalan nyata, mulai dari pemodelan data, analisis statistik, hingga simulasi sistem dinamis.

Pada tugas besar ini, kami diminta mengimplementasikan berbagai metode penyelesaian sistem persamaan linear, perhitungan determinan, pencarian invers matriks, interpolasi polinomial, interpolasi kurva Bézier kubik, dan regresi polinomial menggunakan bahasa pemrograman Java dalam bentuk pustaka (*library*) yang dapat digunakan secara modular dan terdokumentasi dengan baik.

1.2 Tujuan

Adapun tujuan pengerjaan tugas besar ini adalah sebagai berikut:

1. Mengimplementasikan berbagai metode penyelesaian sistem persamaan linear, perhitungan determinan, invers matriks, interpolasi, dan regresi polinomial secara mandiri dalam bahasa Java.
2. Membuat pustaka (*library*) yang dapat digunakan secara modular dan terdokumentasi dengan baik.
3. Mengintegrasikan pustaka yang dibuat ke dalam sebuah program yang dapat menerima masukan dari pengguna dan menampilkan hasilnya dengan format yang jelas.
4. Menguji pustaka yang dibuat pada berbagai kasus uji dan menganalisis hasilnya.

Dengan demikian, tugas besar ini diharapkan dapat menjadi sarana pembelajaran yang efektif dalam memahami dan mengaplikasikan konsep-konsep aljabar linear secara komputasional, serta memberikan bekal kepada kami dengan keterampilan praktis yang sangat dibutuhkan di dunia profesional.

2 Dasar Teori

2.1 Sistem Persamaan Linear

Sistem Persamaan Linear (SPL) adalah himpunan berhingga dari persamaan-persamaan linear dengan variabel-variabel yang sama. Linear berarti pangkat tertinggi di dalam variabelnya adalah satu. Sebuah SPL dengan m buah persamaan dan n variabel $x_1, x_2, \ldots, x_n$ berbentuk:

$$\begin{aligned} a_{11}x_1 + \ a_{12}x_2 + \cdots + \ a_{1n}x_n &= \ b_1 \\ a_{21}x_1 + \ a_{22}x_2 + \cdots + \ a_{2n}x_n &= \ b_2 \\ \vdots \qquad\qquad \ddots \qquad\qquad &= \ \vdots \\ a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n &= b_m \end{aligned}$$

atau dalam bentuk perkalian matriks (kiri) dan matriks *augmented* (kanan).

$$\underbrace{\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix}}_{A} \underbrace{\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}}_{\mathbf{x}} = \underbrace{\begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix}}_{\mathbf{b}} \quad \Longleftrightarrow \quad \left[\begin{array}{cccc|c} a_{11} & a_{12} & \cdots & a_{1n} & b_1 \\ a_{21} & a_{22} & \cdots & a_{2n} & b_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} & b_n \end{array}\right].$$

Solusi dari SPL dapat berupa solusi tunggal, solusi banyak, atau tidak ada solusi. Metode penyelesaian yang umum digunakan antara lain eliminasi Gauss, eliminasi Gauss-Jordan, kaidah Cramer, dan metode matriks balikan.

2.1.1 Eliminasi Gauss

Metode eliminasi Gauss adalah metode yang dikembangkan dari metode eliminasi, yaitu menghilangkan atau mengurangi jumlah variabel sehingga diperoleh nilai dari suatu variabel. Berikut adalah langkah-langkah penyelesaian SPL menggunakan metode eliminasi Gauss.

2.1.1.1 Nyatakan SPL dalam Bentuk Matriks *Augmented*. Matriks *augmented* merupakan bentuk matriks yang menggabungkan matriks koefisien dari suatu SPL dengan vektor konstanta dari persamaan tersebut. Contoh:

$$\begin{aligned} x_1 + 3x_2 - 6x_3 &= 9 \\ 2x_1 - 6x_2 + 4x_3 &= 7 \\ 5x_1 + 2x_2 - 5x_3 &= -2 \end{aligned}$$

diubah menjadi matriks *augmented*

$$\begin{bmatrix} 1 & 3 & -6 & 9 \\ 2 & -6 & 4 & 7 \\ 5 & 2 & -5 & -2 \end{bmatrix}$$

2.1.1.2 Terapkan Operasi Baris Elementer (OBE).

OBE diterapkan pada matriks *augmented* sampai terbentuk matriks eselon baris. OBE adalah suatu operasi untuk mengubah matriks koefisien menjadi bentuk segitiga atau bentuk eselon baris. Bentuk matriks eselon baris adalah matriks yang memiliki 1 utama (*leading one*) pada setiap baris, kecuali pada baris yang seluruhnya nol. Sifat-sifat matriks eselon baris yakni jika sebuah baris tidak terdiri dari seluruhnya nol, maka bilangan tidak nol pertama di dalam baris tersebut adalah 1 (1 utama), jika ada baris yang seluruhnya nol, maka semua baris itu dikumpulkan pada bagian bawah matriks, dan di dalam dua baris berurutan yang tidak seluruhnya nol, 1 utama pada baris yang lebih rendah terdapat lebih jauh ke kanan daripada 1 utama pada baris yang lebih tinggi. Berikut adalah contoh dari matriks eselon baris.

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

Terdapat tiga operasi baris elementer terhadap matriks *augmented*. Pertama, kalikan sebuah baris dengan konstanta tidak nol. Kedua, pertukarkan dua buah baris. Ketiga, tambahkan sebuah baris dengan kelipatan baris lainnya.

2.1.1.3 Penyulihan mundur.

Selesaikan persamaan yang berkoresponden pada matriks eselon baris dengan teknik penyulihan mundur.

2.1.2 Eliminasi Gauss-Jordan

Metode eliminasi Gauss-Jordan merupakan pengembangan dari metode eliminasi Gauss. Operasi baris elementer (OBE) akan diterapkan pada matriks *augmented* sehingga menghasilkan matriks eselon baris tereduksi. Matriks eselon baris tereduksi memiliki ciri yakni memiliki semua nilai nol pada kolom di bawah dan di atas 1 utama. Contoh matriks eselon baris tereduksi adalah sebagai berikut.

$$\begin{bmatrix} 1 & * & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad \text{atau} \quad \begin{bmatrix} 0 & 1 & 0 & 0 & * \\ 0 & 0 & 1 & 0 & * \\ 0 & 0 & 0 & 1 & * \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Tidak diperlukan lagi substitusi secara mundur untuk memperoleh nilai-nilai variabel sebab nilai variabel langsung diperoleh dari matriks *augmented* akhir (jika solusinya unik). Metode ini terdiri dari dua fase: fase maju (*forward phase*) atau fase eliminasi Gauss dan fase mundur yang menghasilkan nilai-nilai 0 di atas satu utama.

2.1.3 Kaidah Cramer

Kaidah Cramer berbunyi: jika A$\mathbf{x} = \mathbf{b}$ adalah SPL yang terdiri dari n persamaan linear dengan n peubah sedemikian sehingga $\det(A) \neq 0$, maka SPL tersebut memiliki solusi yang unik yaitu

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

yang dalam hal ini, A_j adalah matriks yang diperoleh dengan mengganti entri pada kolom ke-j dari A dengan entri dari matriks

$$\mathbf{b} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix}.$$

2.1.4 Metode Matriks Balikan

Sistem Persamaan Linear (SPL) dapat dinyatakan dalam bentuk $\mathrm{A}\mathbf{x} = \mathbf{b}$. Kalikan kedua ruas persamaan dengan A^{-1}:

$$(\mathrm{A}^{-1})\,\mathrm{A}\mathbf{x} = (\mathrm{A}^{-1})\,\mathbf{b} \quad \Rightarrow \quad \mathbf{I}\,\mathbf{x} = \mathrm{A}^{-1}\mathbf{b} \quad \Rightarrow \quad \mathbf{x} = \mathrm{A}^{-1}\mathbf{b}$$

Jadi, solusi SPL $\mathrm{A}\mathbf{x} = \mathbf{b}$ adalah $\mathbf{x} = \mathrm{A}^{-1}\mathbf{b}$.

2.2 Determinan Matriks

Determinan adalah konsep dalam matematika yang digunakan untuk mengukur sifat-sifat matriks yang merupakan susunan bilangan dalam bentuk persegi. Determinan sebuah matriks dapat dihitung dengan dua metode, yaitu metode ekspansi kofaktor dan metode reduksi baris.

2.2.1 Metode Ekspansi Kofaktor

Misalkan A adalah matriks berukuran $n \times n$,

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

didefinisikan bahwa M_{ij} merupakan *minor* entri a_{ij} atau determinan sub-matriks yang elemen-elemennya tidak berada pada baris i dan kolom j, sedangkan C_{ij} merupakan $(-1)^{i+j}M_{ij}$ yang disebut sebagai *kofaktor* entri a_{ij}. Dengan menggunakan kofaktor, determinan matriks dapat dihitung dengan salah satu dari persamaan berikut (baik secara baris maupun kolom).

$$\begin{aligned} \det(A) &= a_{11}C_{11} + a_{12}C_{12} + \cdots + a_{1n}C_{1n} \\ \det(A) &= a_{21}C_{21} + a_{22}C_{22} + \cdots + a_{2n}C_{2n} \\ \det(A) &= a_{n1}C_{n1} + a_{n2}C_{n2} + \cdots + a_{nn}C_{nn} \end{aligned}$$

$$\begin{aligned} \det(A) &= a_{11}C_{11} + a_{21}C_{21} + \cdots + a_{n1}C_{n1} \\ \det(A) &= a_{12}C_{12} + a_{22}C_{22} + \cdots + a_{n2}C_{n2} \\ \det(A) &= a_{1n}C_{1n} + a_{2n}C_{2n} + \cdots + a_{nn}C_{nn} \end{aligned}$$

2.2.2 Metode Reduksi Baris

Misalkan A adalah matriks berukuran $n \times n$, determinan matriks A dapat diperoleh dengan melakukan OBE pada matriks A sampai diperoleh matriks segitiga (segitiga bawah atau atas). Maka,

$$\det(A) = (-1)^p \, a'_{11}a'_{22} \cdots a'_{nn}$$

dengan p menyatakan banyaknya operasi pertukaran baris di dalam OBE. Jika selama reduksi baris ada OBE berupa perkalian baris-baris matriks dengan $k_1, k_2, \ldots, k_m$, maka

$$\det(A) = \frac{(-1)^p \, a'_{11}a'_{22} \cdots a'_{nn}}{k_1 k_2 \cdots k_m}.$$

2.3 Invers Matriks

Misalkan A adalah matriks berukuran $n \times n$, balikan (inverse) matriks A adalah A^{-1} sedemikian sehingga $AA^{-1} = A^{-1}A = I$. Untuk mencari balikan dari sebuah matriks, terdapat dua metode yang dapat digunakan yakni metode matriks adjoin dan metode eliminasi Gauss-Jordan.

2.3.1 Metode Matriks Adjoin

Misalkan A adalah matriks berukuran $n \times n$, balikan matriks A dapat dihitung dengan menggunakan rumus berikut.

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

Adj(A) merupakan *transpose* matriks kofaktor. Matriks kofaktor sendiri berbentuk seperti berikut,

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

dengan C_{ij} adalah kofaktor entri a_{ij}.

2.3.2 Metode *Augment*

Metode *Augment* dapat digunakan untuk menghitung matriks balikan. Untuk matriks A yang berukuran $n \times n$, matriks balikannya, yaitu A^{-1}, dapat dicari dengan cara $[A \mid I]$ lalu terapkan Gauss-Jordan dan menghasilkan $[I \mid A^{-1}]$ yang dalam hal ini I adalah matriks identitas berukuran $n \times n$. Metode *Augment* diterapkan secara simultan untuk A maupun I.

2.4 Interpolasi Polinomial

Interpolasi polinomial adalah metode untuk menentukan suatu polinomial $P_n(x)$ berderajat n yang melalui $n+1$ titik data $(x_0, y_0), (x_1, y_1), \ldots, (x_n, y_n)$, x_i diurutkan dengan x_0 terkecil dan x_n terbesar. Secara umum, polinomial tersebut dapat dituliskan sebagai berikut.

$$P_n(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_n x^n$$

dengan koefisien $a_0, a_1, a_2, \ldots, a_n$ yang ditentukan sedemikian rupa sehingga $P_n(x_{\text{perkiraan}}) = y_{\text{perkiraan}}$ untuk semua nilai $x_0 < x_{\text{perkiraan}} < x_n$. Derajat persamaan polinom ditentukan oleh jumlah titik data yang diberikan dikurangi satu.

Koefisien polinomial dapat dicari dengan menyulihkan titik-titik data ke dalam persamaan polinomial dan menyelesaikan sistem persamaan linear yang dihasilkan dengan metode eliminasi Gauss. SPL dapat disusun sebagai berikut.

$$\begin{cases} a_0 + a_1 x_0 + a_2 x_0^2 + \cdots + a_n x_0^n = y_0 \\ a_0 + a_1 x_1 + a_2 x_1^2 + \cdots + a_n x_1^n = y_1 \\ \vdots \\ a_0 + a_1 x_n + a_2 x_n^2 + \cdots + a_n x_n^n = y_n \end{cases}$$

Persamaanpersamaan di atas dapat dipetakan menjadi sebuah permasalahan sistem persamaan linear sehingga dapat diselesaikan dengan menggunakan metode eliminasi Gauss untuk menyelesaikan sistem tersebut dan menemukan koefisien polinomial $a_0, a_1, a_2, \ldots, a_n$. Dari polinom $P_n(x)$ yang dibentuk dengan koefisien tersebut, nilai y dapat diperkirakan untuk nilai x yang berada di antara titik-titik data yang diberikan (interpolasi) maupun di luar titik-titik data tersebut (ekstrapolasi).

2.5 Interpolasi Splina Bézier Kubik

Kurva Bézier Kubik adalah kurva parametrik yang didefinisikan oleh empat titik kontrol yang mempengaruhi bentuk kurva tersebut. Titik kontrol pertama (P_0) dan terakhir (P_3) adalah titik awal dan akhir kurva, sedangkan titik-titik kontrol lainnya (P_1 dan P_2) menentukan arah dan kelengkungan kurva. Secara matematis, kurva Bézier Kubik dapat ditulis sebagai persamaan parametrik

$$B(t) = (1-t)^3 P_0 + 3t(1-t)^2 P_1 + 3t^2(1-t) P_2 + t^3 P_3, \quad 0 \le t \le 1$$

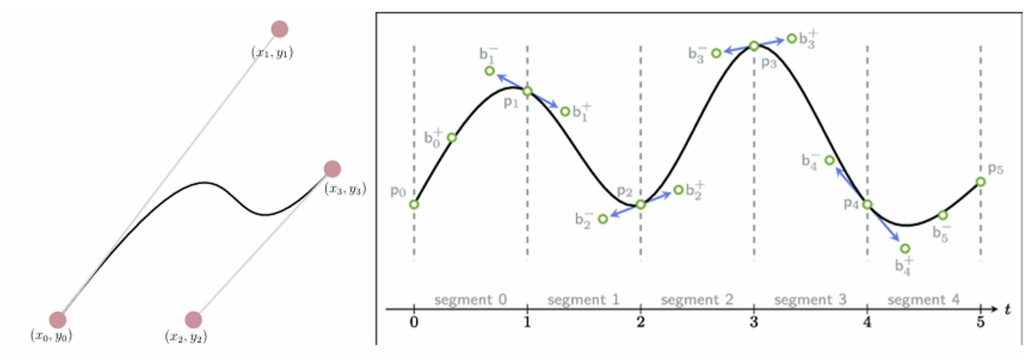

Figure 2: Kurva Bézier Kubik dengan empat titik kontrol (kiri) dan kurva komposit yang tersusun atas banyak splina Bézier Kubik (kanan).

Misal diberikan titik-titik interpolasi $S_0, S_1, \ldots, S_n$ dan perlu menemukan titik-titik kontrol $B_0, B_1, \ldots, B_n$ yang sesuai untuk membangun kurva halus yang melewati semua titik S_i. Kita dapat membangun kurvanya dari splina Bézier Kubik, yakni himpunan kurva Bézier Kubik.

Masalah ini dapat direduksi menjadi penyelesaian SPL yang didapatkan dari hubungan antara titik-titik S_i dan titik-titik kontrol B_i. Sistem persamaan tersebut adalah sebagai berikut.

$$\begin{cases} 4\mathbf{b}_1 + \mathbf{b}_2 = 6\mathbf{s}_1 - \mathbf{s}_0 \\ \mathbf{b}_1 + 4\mathbf{b}_2 + \mathbf{b}_3 = 6\mathbf{s}_2 \\ \mathbf{b}_2 + 4\mathbf{b}_3 + \mathbf{b}_4 = 6\mathbf{s}_3 \\ \cdots \\ \mathbf{b}_{n-2} + 4\mathbf{b}_{n-1} = 6\mathbf{s}_{n-1} - \mathbf{s}_n \end{cases}$$

dengan b_k dan s_k adalah vektor posisi dari titik-titik kontrol B_k dan titik-titik interpolasi S_k, atau dapat dinyatakan dalam bentuk perkalian matriks,

$$\begin{bmatrix} 4 & 1 & & & \\ 1 & 4 & 1 & & \\ & 1 & 4 & \ddots & \\ & & \ddots & \ddots & 1 \\ & & & 1 & 4 \end{bmatrix} \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ \vdots \\ b_{n-1} \end{bmatrix} = \begin{bmatrix} 6s_1 - s_0 \\ 6s_2 \\ 6s_3 \\ \vdots \\ 6s_{n-1} - s_n \end{bmatrix}.$$

Karena vektor posisi b_k dan s_k memiliki dua komponen (x dan y), sistem persamaan di atas dapat dipisahkan menjadi dua sistem persamaan linear yang terpisah, satu untuk komponen x dan satu untuk komponen y. Dengan menyelesaikan kedua sistem persamaan tersebut, kita dapat menentukan posisi titik-titik kontrol B_k yang diperlukan untuk membentuk kurva Bézier kubik yang halus dan sesuai dengan titik-titik interpolasi yang diberikan.

2.6 Regresi Polinomial Berganda (*Multivariate Polynomial Regression*)

Diberikan n titik data $(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)$ dengan x_i adalah variabel dan y_i hasil pengukuran ke-i $(1 \le i \le n)$, regresi linear sederhana bertujuan untuk menemukan garis lurus

$$y = \beta_0 + \beta_1 x + \varepsilon$$

yang paling sesuai dengan data, dengan ε adalah galatnya. Namun, dalam banyak kasus, besaran yang diamati bisa dipengaruhi oleh lebih dari satu variabel. Untuk mengatasi hal ini, dapat digunakan regresi linear berganda

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \varepsilon,$$

dengan ε adalah galat yang mengikuti distribusi normal dengan rata-rata nol dan varians konstan. Tugas kita adalah mengestimasi nilai koefisien $\beta_0, \beta_1, \beta_2$, dan β_3, berdasarkan data yang diberikan. Secara umum, andai kata ada n variabel independen $x_1, \dots, x_n$, maka persamaan regresi linear berganda dapat dituliskan sebagai

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_n x_n + \varepsilon,$$

sehingga jika kita memiliki m titik data dan x_{ij} nilai variabel x_j pada titik data ke-i, serta y_i dan ε_i adalah nilai hasil pengukuran dan galatnya pada titik data ke-i, sistem persamaan linearnya adalah

$$\underset{m\times 1}{\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix}} = \underset{m\times(n+1)}{\begin{bmatrix} 1 & x_{11} & \cdots & x_{1n} \\ 1 & x_{21} & \cdots & x_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_{m1} & \cdots & x_{mn} \end{bmatrix}} \underset{(n+1)\times 1}{\begin{bmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_n \end{bmatrix}} + \underset{m\times 1}{\begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_m \end{bmatrix}},$$

atau secara matriks

$$\mathbf{Y} = X\boldsymbol{\beta} + \boldsymbol{\varepsilon},$$

dengan $\mathbf{Y}$ adalah vektor hasil pengukuran, X matriks desain, $\boldsymbol{\beta}$ vektor koefisien, dan $\boldsymbol{\varepsilon}$ vektor galat.

Tugasnya adalah mengestimasi nilai vektor koefisien $\hat{\boldsymbol{\beta}}$ dengan galat sekecil mungkin. Metode kuadrat terkecil (*least squares*) meminimalkan $\|\boldsymbol{\varepsilon}\|^2 = \|\mathbf{y} - X\boldsymbol{\beta}\|^2$ dan menghasilkan persamaan normal

$$X^\mathsf{T} X\, \hat{\boldsymbol{\beta}} = X^\mathsf{T}\mathbf{y}, \qquad \text{atau} \qquad \hat{\boldsymbol{\beta}} = (X^\mathsf{T} X)^{-1} X^\mathsf{T}\mathbf{y}. \tag{1}$$

Jika hubungan yx tidak linear, gunakan regresi polinomial berganda. Idenya masih mirip: mencari fungsi polinomial yang kurvanya paling sesuai dengan data yang diberikan. Namun, variabel interaksi (hasil kali antar-variabel) dan pangkat yang lebih tinggi turut disertakan. Sebagai contoh, untuk tiga variabel independen x_1, x_2, x_3 dan derajat 3:

$$\begin{aligned} y = \beta_0 &+ \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \beta_4 x_1^2 + \beta_5 x_1 x_2 + \beta_6 x_1 x_3 + \beta_7 x_2^2 + \beta_8 x_2 x_3 + \beta_9 x_3^2 \\ &+ \beta_{10} x_1^3 + \beta_{11} x_1^2 x_2 + \beta_{12} x_1^2 x_3 + \beta_{13} x_1 x_2^2 + \beta_{14} x_1 x_2 x_3 + \beta_{15} x_1 x_3^2 \\ &+ \beta_{16} x_2^3 + \beta_{17} x_2^2 x_3 + \beta_{18} x_2 x_3^2 + \beta_{19} x_3^3 + \varepsilon. \end{aligned}$$

Secara umum, untuk k variabel independen dan derajat polinomial d, jumlah total suku adalah

$$p = \binom{d+k}{k}.$$

Koefisien $\beta_0, \beta_1, \ldots, \beta_p$ dapat dicari dengan mengadopsi pendekatan serupa dengan yang digunakan pada regresi linear berganda. Kali ini, variabel-variabel seperti x_1^2, $x_1 x_2$, x_1^3, dan sebagainya akan dianggap sebagai variabel independen yang berbeda. Dengan demikian, kita dapat menyusun sistem persamaan linear yang mirip dengan yang sebelumnya, tetapi dengan lebih banyak kolom pada matriks desain X untuk mengakomodasi semua suku dalam polinomial. Setelah itu, kita dapat menggunakan metode kuadrat terkecil untuk menemukan estimasi koefisien $\hat{\boldsymbol{\beta}}$ (persamaannya sama persis seperti pada regresi linear berganda). Misalnya, jika kita memiliki lima sampel data dan tiga variabel independen ($k = 3$) dan ingin membuat polinomial dengan derajat dua ($d = 2$), maka jumlah total suku dalam polinomial tersebut adalah

$$p = \binom{d+k}{k} = \binom{2+3}{3} = 10.$$

dan matriks desain X akan berbentuk seperti berikut.

$$X = \begin{bmatrix} 1 & x_{11} & x_{12} & x_{13} & x_{11}^2 & x_{11}x_{12} & x_{11}x_{13} & x_{12}^2 & x_{12}x_{13} & x_{13}^2 \\ 1 & x_{21} & x_{22} & x_{23} & x_{21}^2 & x_{21}x_{22} & x_{21}x_{23} & x_{22}^2 & x_{22}x_{23} & x_{23}^2 \\ 1 & x_{31} & x_{32} & x_{33} & x_{31}^2 & x_{31}x_{32} & x_{31}x_{33} & x_{32}^2 & x_{32}x_{33} & x_{33}^2 \\ 1 & x_{41} & x_{42} & x_{43} & x_{41}^2 & x_{41}x_{42} & x_{41}x_{43} & x_{42}^2 & x_{42}x_{43} & x_{43}^2 \\ 1 & x_{51} & x_{52} & x_{53} & x_{51}^2 & x_{51}x_{52} & x_{51}x_{53} & x_{52}^2 & x_{52}x_{53} & x_{53}^2 \end{bmatrix},$$

yang sebenarnya analog dengan matriks desain untuk regresi linear berganda

$$X = \begin{bmatrix} 1 & x_{11} & x_{12} & x_{13} & x_{14} & x_{15} & x_{16} & x_{17} & x_{18} & x_{19} \\ 1 & x_{21} & x_{22} & x_{23} & x_{24} & x_{25} & x_{26} & x_{27} & x_{28} & x_{29} \\ 1 & x_{31} & x_{32} & x_{33} & x_{34} & x_{35} & x_{36} & x_{37} & x_{38} & x_{39} \\ 1 & x_{41} & x_{42} & x_{43} & x_{44} & x_{45} & x_{46} & x_{47} & x_{48} & x_{49} \\ 1 & x_{51} & x_{52} & x_{53} & x_{54} & x_{55} & x_{56} & x_{57} & x_{58} & x_{59} \end{bmatrix},$$

dengan x_{ij} adalah nilai dari suku ke-j pada titik data ke-i. Pada intinya, cukup gunakan persamaan ke (1) untuk menemukan estimasi koefisien $\hat{\beta}$.

3 Implementasi

3.1 Struktur Program

Adapun struktur dari pengembangan program ini dapat dilihat pada diagram struktur direktori berikut.

```
├── bin
├── docs
├── src
│   └── main
│       ├── java
│       │   └── algeo
│       │       ├── modules
│       │       └── utils
│       └── resources
├── target
│   ├── classes
│   │   ├── algeo
│   │   │   ├── modules
│   │   │   └── utils
│   ├── generated-sources
│   │   └── annotations
│   └── maven-status
│       └── maven-compiler-plugin
│           └── compile
│               └── default-compile
└── test
    ├── input
    └── output
```

Figure 3: Struktur Direktori

Struktur ini mengikuti standar proyek Java berbasis Maven, di mana:

1. `bin` berisi *file* biner hasil kompilasi yang dapat dijalankan langsung.
2. `docs` berisi dokumentasi proyek, seperti laporan tugas besar.
3. `src/main/java` berisi kode sumber utama program, yang dikelompokkan dalam paket algeo.
    - `algeo/modules` berisi implementasi kelas-kelas utama dan metode perhitungan utama program (misalnya penyelesaian SPL, determinan, invers, interpolasi, dan regresi).
    - `algeo/utils` berisi kelas-kelas pendukung yang berfungsi sebagai *helper* atau *controller* untuk membantu proses utama dalam program.
4. `src/main/resources` berisi *file* sumber daya tambahan yang mendukung program, seperti *file* konfigurasi atau aset antarmuka pengguna *(GUI)*.

5. `target` berisi hasil kompilasi, *file build*, serta keluaran lain yang dihasilkan oleh Maven.
    - `classes` berisi hasil kompilasi dari kode sumber utama.
    - `generated-sources` menyimpan *file* sumber yang dihasilkan secara otomatis oleh proses *build*.
    - `maven-status/maven-compiler-plugin/compile/default-compile` berisi metadata status kompilasi dari Maven.
6. `test` digunakan untuk keperluan pengujian program.
    - `input` menyimpan data uji masukan.
    - `output` menyimpan hasil keluaran dari proses pengujian.

Dengan struktur tersebut, pengembangan program menjadi lebih terorganisasi, modular, dan mudah untuk dikelola, diuji, serta dikembangkan lebih lanjut.

3.2 Kelas-Kelas Program

Berikut adalah deskripsi dan atribut serta operasi pada kelas-kelas pada modul yang digunakan di program ini.

3.2.1 *Class Matrix*

Class Matrix merupakan kelas yang mengurus terkait matrix yang digunakan meliputi atribut utama matriks, konstruktor, selektor, dan metode.

- **Atribut**

No.	Nama Atribut	Tipe Data	Deskripsi
1.	baris	integer	Atribut yang memberikan jumlah baris matriks
2.	kolom	integer	Atribut yang memberikan jumlah kolom matriks
3.	data	double[][]	Atribut yang memberikan isi matriks

- **Metode**

No.	Nama Metode	Tipe Data	Parameter	Deskripsi
1.	Matrix	public	integer, integer	Konstruktor membuat sebuah matriks kosong dengan parameter jumlah baris dan kolomnya.
2.	Matrix	public	double[][]	Konstruktor membuat sebuah matriks dari masukan sebuah *array* 2 dimensi
3.	Matrix	public	Matrix	Konstruktor membuat sebuah matriks dari matriks yang sudah ada

4.	getBaris	public int		Selektor yang mengembalikan jumlah baris dari objek matriks
5.	getKolom	public int		Selektor yang mengembalikan jumlah kolom dari objek matriks
6.	getData	public double[][]		Selektor yang mengembalikan seluruh isi matriks dalam bentuk *array* 2 dimensi
7.	getElmt	public double	integer, integer	Selektor yang mengembalikan sebuah elemen dari matriks pada baris dan kolom tertentu
8.	setElmt	public void	integer, integer, double	Metode yang mengubah nilai sebuah elemen pada matriks dengan baris dan kolom tertentu
9.	copy	public Matrix		Metode yang dapat menyalin isi data pada sebuah objek matriks
10.	penjumlahan	public Matrix	Matrix	Metode untuk menjumlahkan sebuah objek matriks dengan matriks tertentu
11.	pengurangan	public Matrix	Matrix	Metode untuk menjumlahkan sebuah objek matriks dengan matriks tertentu
12.	perkalian	public Matrix	double	Metode untuk mengalikan sebuah objek matriks dengan nilai skalar
13.	perkalian	public Matrix	Matrix	Metode untuk mengalikan sebuah objek matriks dengan suatu matriks
14.	gantiKolom	public Matrix	double[], integer	Metode untuk mengganti kolom sebuah objek matriks dengan masukan sebuah kolom baru

15.	ambilMatrixKoef	public Matrix		Selektor untuk mengambil sebuah matriks koefisien dari objek matriks
16.	ambilMatrixKons	public double[]		Selektor untuk mengambil sebuah matriks konstanta dari objek matriks (kolom terakhir)
17.	transpose	public Matrix		Metode untuk mendapatkan *transpose* dari objek matriks
18.	getMinorEntri	public Matrix	integer, integer	Selektor untuk mendapatkan sebuah matriks *minor entry* dari suatu baris dan kolom
19.	kofaktorMatrix	public Matrix		Metode untuk mendapatkan matriks kofaktor dari sebuah matriks
20.	adjoin	public Matrix		Metode untuk mendapatkan nilai *adjoin* objek matriks
21.	hitungLead	public int		Metode untuk menghitung nilai *leading non-zero* dari reduksi Gauss objek matriks
22.	print	public void		Metode untuk mencetak sebuah objek matriks

3.2.2 *Class Determinan*

Class Determinan merupakan kelas yang menangani proses perhitungan determinan matriks. Pada implementasi program ini, tersedia dua metode perhitungan determinan, yaitu menggunakan Reduksi Baris (OBE) dan Ekspansi Kofaktor.

No.	Nama Metode	Tipe Data	Deskripsi
1.	determinanOBE()	Public static double	Metode ini menghitung determinan matriks dengan pendekatan reduksi baris (Operasi Baris Elementer / OBE).

2.	determinanEkspansiKofaktor()	Public static double	Metode ini menghitung determinan matriks dengan menggunakan konsep ekspansi kofaktor.

3.2.3 *Class Invers*

Class Invers merupakan kelas yang menangani proses perhitungan invers matriks. Pada implementasi program ini, tersedia dua metode perhitungan invers, yaitu menggunakan metode Augment dan metode Adjoin.

No.	Nama Metode	Tipe Data	Deskripsi
1.	inversAdjoin()	public static double	Metode ini menghitung invers matriks dengan pendekatan metode adjoin matriks.
2.	inversAugment()	public static double	Metode ini menghitung invers matriks dengan pendekatan metode matriks *augmented*.

3.2.4 *Class SPL*

Class SPL merupakan kelas yang menyelesaikan sistem persamaan linear. Pada implementasi program ini, terdapat empat metode yang digunakan, yakni metode Eliminasi Gauss, Eliminasi Gauss-Jordan, Kaidah Cramer, dan Matriks Balikan (Invers).

No.	Nama Metode	Tipe Data	Deskripsi
1.	splGauss()	public static double	Metode ini menyelesaikan sistem persamaan linear menggunakan konsep Gauss.
2.	splGaussJordan()	public static double	Metode ini menyelesaikan sistem persamaan linear menggunakan konsep Gauss-Jordan.
3.	splCramer()	public static double	Metode ini menyelesaikan sistem persamaan linear menggunakan konsep kaidah Cramer.
4.	splInvers()	public static double	Metode ini menyelesaikan sistem persamaan linear menggunakan konsep matriks balikan (Invers).

3.2.5 *Class Interpolasi*

Class Interpolasi merupakan kelas yang digunakan untuk melakukan perhitungan Interpolasi Polinomial dan Splina (Bézier) berdasarkan data titik (x, y). Tujuannya adalah untuk mendapatkan fungsi atau kurva yang melewati titik-titik data tertentu, sehingga bisa digunakan untuk memperkirakan (estimasi) nilai di antara titik-titik tersebut.

No.	Nama Metode	Tipe Data	Parameter	Deskripsi

1.	interpolasiPolinomial()	public static Matrix	Matrix	Metode ini digunakan untuk mencari koefisien dari polinomial interpolasi berdasarkan titik-titik data x dan y yang diberikan.
2.	evaluasiPolinomial()	public static double	Matrix, double	Metode ini menghitung nilai hasil interpolasi f(t) berdasarkan koefisien polinomial yang telah diperoleh.
3.	cekXY()	private static void	Matrix	Metode ini berfungsi sebagai validasi awal terhadap data input.
4.	interpolasiSplinaBezier()	public static Matrix	Matrix	Metode ini digunakan untuk membentuk titik-titik kurva halus (Bézier splina).
5.	tridiagonal()	private static double[]	double[], double[], double[], double[]	Metode ini adalah *helper* untuk menyelesaikan sistem persamaan linear tridiagonal, yang sering muncul pada interpolasi splina.

3.2.6 *Class RegresiPolinomGanda*

Class RegresiPolinomGanda merupakan kelas yang berfungsi untuk menyelesaikan perhitungan persamaan polinomial menggunakan metode regresi. Pada implementasi program ini, terdapat enam metode utama yang digunakan untuk membantu proses perhitungan sesuai dengan spesifikasi tugas.

No.	Nama Metode	Tipe Data	Parameter	Deskripsi
1.	fiturPolinom()	public static Matrix	Matrix, int	Metode ini membangun fitur polinomial sesuai derajat yang ditentukan dalam bentuk matriks.

2.	generateExponents()	public static List<int[]>	int, int	Metode ini menyimpan kombinasi variabel beserta pangkatnya yang digunakan dalam pembentukan fitur.
3.	generateRecursively()	private static void	List<int[]>, int[], int, int	Metode ini adalah sebuah rekursifias yang membantu pembentukan kombinasi variabel (pada *generateExponents*).
4.	regresiPolinom()	public static Matrix	Matrix, double[], int	Metode ini menghitung koefisien regresi polinomial (β) berdasarkan data masukan x dan y.
5.	bentukPersamaan()	public static String	Matrix, int, int	Metode ini mengonversi koefisien hasil regresi menjadi bentuk persamaan matematis y(x).
6.	hitungY()	public static double	Matrix, double[], int	Metode ini menghitung nilai prediksi terhadap y berdasarkan nilai variabel input x.

3.2.7 *Class SPLController*

Class SPLController merupakan kelas yang berfungsi untuk menampilkan perhitungan Sistem Persamaan Linear pada *Graphical User Interface* (GUI). Pada implementasi program ini, terdapat tujuh metode utama yang digunakan dalam merealisasikan program.

No.	Nama Metode	Tipe Data	Deskripsi
1.	setMatrix()	public void	Metode untuk menginisiasikan matriks.
2.	eliminasiGauss()	public void	Metode untuk melakukan perhitungan SPL dengan metode eliminasi Gauss.
3.	eliminasiGaussJordan()	public void	Metode untuk melakukan perhitungan SPL dengan metode eliminasi Gauss-Jordan.
4.	kaidahCramer()	public void	Metode untuk melakukan perhitungan SPL dengan metode kaidah Cramer.
5.	metodeMatriksBalikan()	public void	Metode untuk mengimplementasikan balikan dari matriks.
6.	switchToDisplayStep()	public void	Metode untuk menampilkan setiap langkah perhitungan SPL.
7.	switchToMenu()	public void	Metode untuk berganti ke menu.

3.2.8 *Class DetController*

Class DetController merupakan kelas yang berfungsi untuk menampilkan perhitungan determinan matriks pada *Graphical User Interface* (GUI). Pada implementasi program ini, terdapat lima metode utama yang digunakan dalam merealisasikan program.

No.	Nama Metode	Tipe Data	Deskripsi
1.	setMatrix()	public void	Metode untuk menginisiasikan matriks.
2.	metodeEkspansiKofaktor()	public void	Metode untuk menghitung determinan matriks menggunakan metode ekspansi kofaktor.
3.	metodeReduksiBaris()	public void	Metode untuk menghitung determinan matriks menggunakan metode reduksi baris.
4.	switchToDisplayStep()	public void	Metode untuk menampilkan setiap langkah perhitungan determinan.
5.	switchToMenu()	public void	Metode untuk berganti ke menu.

3.2.9 *Class InvController*

Class InvController merupakan kelas yang berfungsi untuk menampilkan perhitungan invers matriks pada *Graphical User Interface* (GUI). Pada implementasi program ini, terdapat lima metode utama yang digunakan dalam merealisasikan program.

No.	Nama Metode	Tipe Data	Deskripsi
1.	setMatrix()	public void	Metode untuk menginisiasikan matriks.
2.	metodeAugment()	public void	Metode untuk melakukan perhitungan invers matriks menggunakan metode *augmentasi.*
3.	metodeAdjoin()	public void	Metode untuk melakukan perhitungan invers matriks menggunakan metode adjoin.
4.	switchToDisplayStep()	public void	Metode untuk menampilkan setiap langkah perhitungan invers.
5.	switchToMenu()	public void	Metode untuk berganti ke menu.

3.3 Cara Kerja dan Alur Program

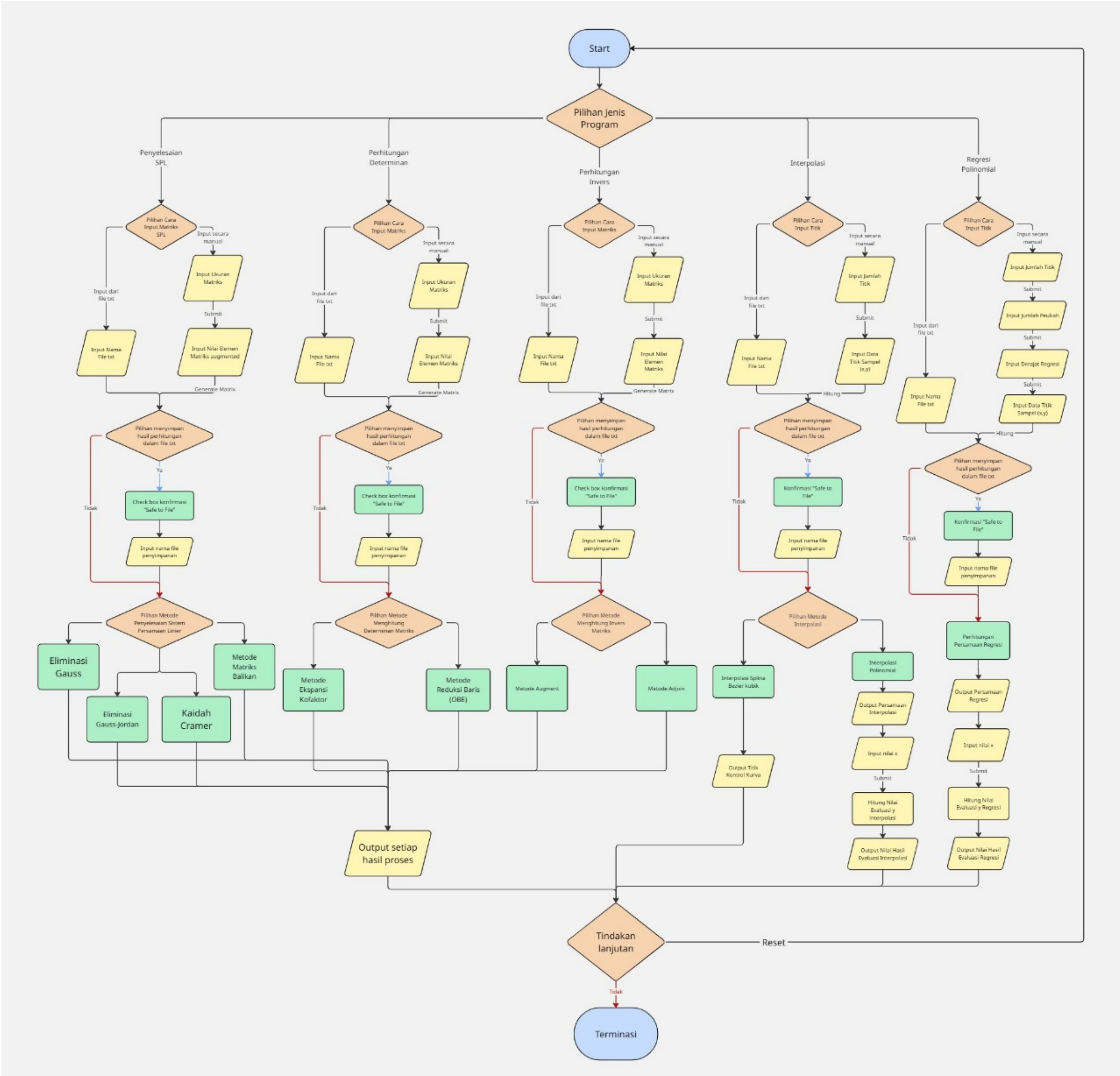

Figure 4: Diagram Alir Program

Secara umum, program ini terdiri atas beberapa fitur utama yang mencakup:

1. Penyelesaian Sistem Persamaan Linear (SPL)
2. Perhitungan Determinan Matriks
3. Perhitungan Invers Matriks
4. Interpolasi
5. Regresi Polinomial Berganda

Program dapat menerima input data melalui dua cara, yaitu melalui *file* teks (.txt) atau input manual oleh pengguna.

Setiap jenis program memiliki spesifikasi format *input* yang berbeda, seperti ditunjukkan pada tabel berikut:

No.	Program	Format Input File txt	Format Input Manual User

1.	Penyelesaian SPL	Matriks augmented	Jumlah baris (m) dan kolom (n) serta matriks *augmented* (matriks koefisien m x n dengan matriks konstanta)
2.	Perhitungan Determinan	Matriks koefisien m baris x n kolom	Jumlah baris (m) dan kolom (n) serta matriks koefisien m x n.
3.	Perhitungan Invers	Matriks koefisien m baris dan n kolom	Jumlah baris (m) dan kolom (n) serta matriks koefisien m x n.
4.	Interpolasi	Matriks 2 kolom untuk sampel titik (x, y)	Jumlah sampel titik serta titik sampel (x,y).
5.	Regresi Polinomial Berganda	Matriks 2 kolom untuk sampel titik (x,y) dan sebuah nilai derajat	Jumlah sampel titik, titik sampel (x,y), serta nilai derajat pangkat.

Sebelum diproses, format *file* .txt akan divalidasi terlebih dahulu untuk memastikan kesesuaiannya dengan jenis program yang dipilih. Setelah data matriks atau titik sampel berhasil dimasukkan (baik melalui *file* maupun *input* manual), pengguna dapat memilih metode perhitungan yang tersedia sesuai jenis program. Pengguna juga dapat memilih untuk menyimpan hasil perhitungan ke dalam *file* teks (.txt) dengan mencentang *checkbox* konfirmasi dan memasukkan nama *file*. Program kemudian akan menampilkan tahapan proses perhitungan hingga menghasilkan nilai akhir sesuai dengan metode yang digunakan. Setelah perhitungan selesai, pengguna diberikan beberapa pilihan tindakan lanjutan, pengguna dapat memulai ulang program (dengan tekan tombol reset) untuk kembali menggunakan kalkulator mulai dari awal program. Dengan alur kerja tersebut, program diharapkan dapat membantu pengguna dalam melakukan berbagai perhitungan matriks dan analisis data secara efisien, akurat, dan terstruktur.

4 Eksperimen

4.1 Tampilan Awal

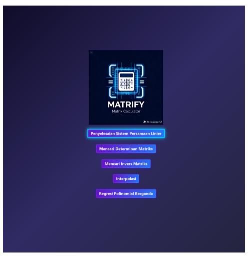

Figure 5: Tampilan Awal

4.2 Metode Penyelesaian SPL

Figure 6: Pilihan Penyelesaian SPL

Figure 7: Eliminasi Gauss Langkah Awal

Figure 8: Eliminasi Gauss Hasil Akhir

Figure 9: Eliminasi Gauss-Jordan Langkah Awal

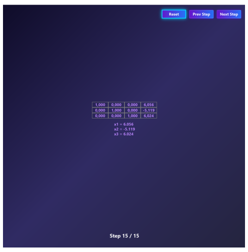

Figure 10: Eliminasi Gauss-Jordan Hasil Akhir

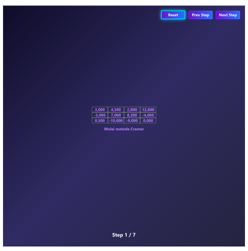

Figure 11: Kaidah Cramer Langkah Awal

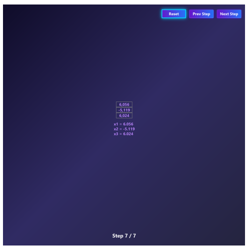

Figure 12: Kaidah Cramer Hasil Akhir

Figure 13: Metode Matriks Balikan Langkah Awal

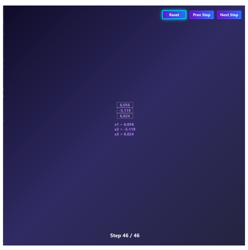

Figure 14: Metode Matriks Balikan Hasil Akhir

4.3 Determinan Matriks

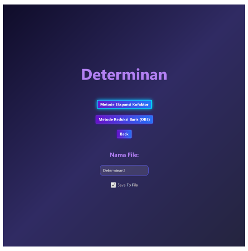

Figure 15: Pilihan Metode Perhitungan Determinan

Figure 16: Metode Ekspansi Kofaktor Langkah Awal

Figure 17: Metode Ekspansi Kofaktor Hasil Akhir

Figure 18: Metode Reduksi Baris Langkah Awal

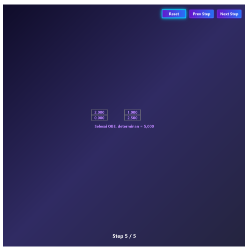

Figure 19: Metode Reduksi Baris Hasil Akhir

4.4 Metode Invers

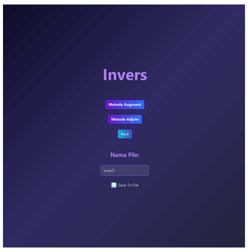

Figure 20: Pilihan Metode Perhitungan Invers

Figure 21: Metode Matriks Adjoin Langkah Awal

Figure 22: Metode Matriks Adjoin Hasil Akhir

Figure 23: Metode Eliminasi Gauss-Jordan Langkah Awal

Figure 24: Metode Eliminasi Gauss-Jordan Hasil Akhir

Figure 25: Matriks yang tidak memiliki balikan

4.5 Interpolasi

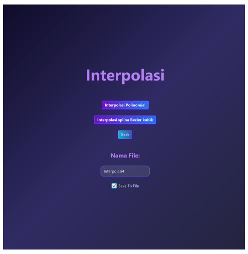

Figure 26: Pilihan Metode Interpolasi

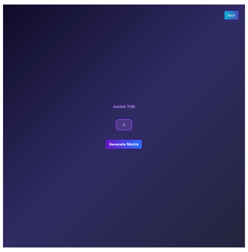

Figure 27: Input Jumlah Titik Interpolasi

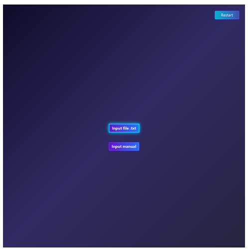

Figure 28: Input Titik Interpolasi

Figure 29: Interpolasi Polinomial Hasil Akhir

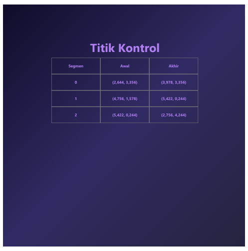

Figure 30: Interpolasi Splina Bézier Kubik

4.6 Regresi Polinomial Berganda

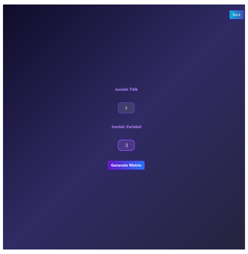

Figure 31: Input Jumlah Titik dan Variabel Regresi Polinomial Berganda

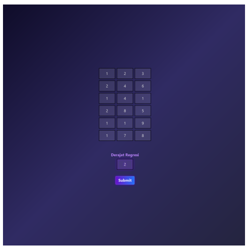

Figure 32: Input Titik Regresi Polinomial Berganda dan Derajat

Figure 33: Hasil Regresi Polinomial Berganda

5 Penutup

Melalui pengerjaan tugas besar ini, kami berhasil memahami serta mengimplementasikan berbagai metode dalam aljabar linear, mulai dari penyelesaian sistem persamaan linear, perhitungan determinan, pencarian invers matriks, hingga interpolasi dan regresi polinomial. Proses ini tidak hanya memperdalam pemahaman teori, tetapi juga melatih keterampilan pemrograman dalam mengembangkan pustaka yang modular, terdokumentasi, dan dapat diuji melalui berbagai kasus uji yang disediakan. Kami menyadari bahwa masih terdapat ruang untuk perbaikan, baik dalam efisiensi implementasi maupun kelengkapan fitur. Oleh karena itu, kami berharap saran dari asisten dapat membantu pengembangan lebih lanjut. Secara keseluruhan, tugas ini memberikan pengalaman berharga dalam mengintegrasikan konsep matematis dengan pemrograman komputasional. Demikian laporan ini kami susun. Semoga dapat bermanfaat dan menjadi bahan evaluasi pembelajaran bagi kami di masa mendatang.

Saran kepada asisten : tolong dipermudah tugasnya, perpanjang waktunya juga ditambah lagi pliss.

"Yok bisa yok kelar, belum tengah semester udah dibantai 4 tubes nih"
"That Interpolasi Bezier, the real bzir dah!"
~Keisha

"Ampun pendiklat!"
~Haris

"Bikin tabel di LaTeX ternyata sangat mudah"
~Helen

6 Daftar Pustaka

6.1 SPL, Determinan, dan Matriks

H. Anton, *Elementary Linear Algebra*, 10th ed. Hoboken, NJ: John Wiley & Sons, 2010. Available: https://archive.org/details/ElementryLinearAlgebraByHowardAnton10thEdition

6.2 Interpolasi Polinomial

G. Muntingh, *Topics in Polynomial Interpolation Theory*, Ph.D. diss., Univ. of Oslo, 2010. Available: https://www.researchgate.net/publication/265819309_Topics_in_Polynomial_Interpolation_Theory

6.3 Interpolasi Splina Bézier Kubik

L. P. Quan and T. A. Nhan, A closed-form solution to the inverse problem in interpolation by a Bézier-spline curve, *Arabian Journal of Mathematics*, 9, 155165, 2020. https://doi.org/10.1007/s40065-019-0241-0

6.4 Regresi Polinomial Berganda

M. Rosenfeld, *Matrix Formulation of OLS: Notes for SOC Methods Project 3*, Stanford University. https://web.stanford.edu/~mrosenfe/soc_meth_proj3/matrix_OLS_NYU_notes.pdf

A. B. Kashlak, *Applied Regression Analysis: Course Notes for STAT 378/502*, Univ. of Alberta, 2022. https://sites.ualberta.ca/~kashlak/data/stat378.pdf

S. Weisberg, *Applied Linear Regression*, 4th ed., Wiley-Blackwell, 2013. https://www.stat.purdue.edu/~qfsong/teaching/525/book/Weisberg-Applied-Linear-Regression-Wiley.pdf

7 Lampiran

7.1 *Link* Github

https://github.com/IRK-23/algeo1-yakin-kau-bung

7.2 *Flowchart* Program

https://miro.com/app/board/uXjVJ9wMf_4=/

7.3 *Drive* Video Pengenalan Program

https://bit.ly/Video-TubesAlgeo1-Kel35-YakinKauBung