LAPORAN TUGAS BESAR

TUGAS BESAR ALJABAR LINEAR DAN GEOMETRI 1

KELOMPOK CHUAKZ

Imanuel Sebastian Girsang (13522058)

Bagas Sambega Rosyada (13522071)

Ellijah Darrellshane Suryanegara (13522097)

DAFTAR ISI

BAB I
DESKRIPSI MASALAH

Matriks adalah suatu tipe data yang berisi kumpulan data yang disusun berdasarkan baris dan kolomnya. Matriks sering disebut sebagai “array dalam array”. Teori dan penggunaan matriks yang sangat luas membuat matriks sering dioperasikan dalam bidang ilmu sains dan rekayasa. Matriks acapkali dijadikan sebagai suatu model yang merepresentasikan kumpulan data terstruktur yang memiliki indeks terurut.

Sistem persamaan linear adalah suatu sistem persamaan yang terdiri dari kumpulan variabel yang membentuk beberapa persamaan (https://en.wikipedia.org/wiki/System_of_linear_equations). Nilai variabel yang ada dalam suatu persamaan linear dapat dicari dengan beberapa cara, beberapa di antaranya adalah dengan metode Gauss, metode Gauss-Jordan, kaidah Cramer, dan menggunakan matriks balikan.

Suatu SPL dapat direpresentasikan dalam suatu matriks yang berbentuk “matriks augmented” (Anton, Howard, 2010). Matriks augmented berukuran mxn terdiri dari m baris yang merepresentasikan m persamaan dan n-1 variabel, dengan kolom paling akhir adalah hasil dari persamaan tersebut. Misalkan suatu SPL terdiri atas:

$$x_1 + x_2 - x_3 = 12$$

$$2x_1 - 4x_2 + 3x_3 = 7$$

$$-x_1 - 2x_3 = 0$$

dapat direpresentasikan dalam bentuk matriks,

$$\begin{bmatrix} 1 & 1 & -1 & 12 \\ 2 & -4 & 3 & 7 \\ -1 & 0 & -2 & 0 \end{bmatrix}$$

Untuk menyelesaikan suatu SPL, dapat dibuat sebuah program Java yang mampu menyelesaikan solusi SPL atau pengolahan matriks menggunakan metode-metode di atas. Dengan menggunakan

BAB II

TEORI SINGKAT

2.1 Eliminasi Gauss

Eliminasi Gauss adalah sebuah teknik untuk menyelesaikan sistem persamaan linear dengan mengubah matriksnya menjadi bentuk matriks eselon baris. Matriks eselon baris memiliki beberapa sifat khusus, yakni:

1. Ketika sebuah baris memiliki angka yang bukan nol, angka pertama yang bukan nol, dan angka tersebut adalah angka 1 maka ia disebut sebagai leading one.
2. Jika ada sebuah baris yang seluruh elemennya adalah 0, baris tersebut akan dipindahkan ke bagian paling bawah dari matriks.
3. Leading one dalam baris yang lebih bawah akan berada di sebelah kanan dari leading one dalam baris di atasnya.

$$\text{(a).}\begin{bmatrix}1 & 0 & 0 & 4\\ 0 & 1 & 0 & 3\\ 0 & 0 & 1 & 2\end{bmatrix} \qquad \text{(c).}\begin{bmatrix}1 & 5 & 0 & 0 & 5 & -1\\ 0 & 0 & 1 & 0 & 3 & 1\\ 0 & 0 & 0 & 1 & 4 & 2\\ 0 & 0 & 0 & 0 & 0 & 0\end{bmatrix}$$

$$\text{(b).}\begin{bmatrix}1 & 0 & 0 & 3 & 2\\ 0 & 1 & 0 & -1 & 4\\ 0 & 0 & 1 & 1 & 2\end{bmatrix} \qquad \text{(d).}\begin{bmatrix}1 & 2 & 0 & 0\\ 0 & 0 & 1 & 0\\ 0 & 0 & 0 & 1\end{bmatrix}$$

Gambar 2.1 Contoh Matriks eselon baris

Untuk mengubah matriks menjadi bentuk matriks eselon baris, diperlukan penggunaan Operasi Baris Elementer (OBE) yang terdiri dari tiga jenis operasi sebagai berikut:

1. Pertukaran dua baris.
2. Pengalian satu baris dengan suatu bilangan yang bukan nol.
3. Penambahan kelipatan suatu baris ke baris lain.

Setelah matriks telah diubah menjadi bentuk eselon baris melalui OBE, akan dihasilkan sejumlah persamaan yang lebih mudah untuk diselesaikan. Dari persamaan-persamaan ini, dapat dilakukan substitusi mundur untuk mendapatkan solusi sistem persamaan linear.

Dalam metode eliminasi Gauss untuk menyelesaikan sistem persamaan linear, terdapat tiga jenis solusi yang mungkin terjadi:

1. Solusi Tunggal (Unique Solution): Terjadi ketika setelah proses eliminasi Gauss, matriks telah diubah menjadi bentuk eselon baris dan tidak ada variabel yang saling terkait atau bergantung. Dalam kasus ini, sistem persamaan linear memiliki satu solusi yang unik yang dapat dihitung secara eksplisit. Hal ini ditandai dengan tidak adanya baris yang

sepenuhnya bernilai 0, ataupun ada konstanta yang berkontradiksi seperti ketika seluruh konstanta memiliki koefisien 0, tetapi hasilnya != 0.

2. Banyak Solusi (Parametrik): Terjadi ketika setelah proses eliminasi Gauss, matriks eselon baris menghasilkan variabel yang saling terkait atau tergantung satu sama lain. Hal ini dapat ditandakan dengan adanya baris yang seluruhnya bernilai 0.
3. Tidak ada solusi: Terjadi ketika setelah eliminasi Gauss, terdapat kontradiksi atau persamaan yang tidak konsisten, seperti 0 = 1. Dalam kasus ini, sistem persamaan linear tidak memiliki solusi yang memenuhi persamaan-persamaannya.

2.2 Eliminasi Gauss-Jordan

Eliminasi Gauss Jordan adalah sebuah metode untuk menemukan solusi dari sistem persamaan linear dengan mengubah matriks menjadi bentuk matriks eselon baris tereduksi. Matriks eselon baris tereduksi memiliki sifat yang sama dengan matriks eselon baris, yaitu setiap baris yang bukan nol memiliki leading one, dan leading one berada di sebelah kanan dari leading one pada baris sebelumnya. Selain itu, matriks eselon baris tereduksi juga memiliki sifat bahwa setiap kolom yang memiliki leading one tidak memiliki angka lain selain nol di kolom tersebut.

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

Gambar 2.2. Contoh matriks eselon baris tereduksi

Untuk mengubah matriks menjadi bentuk matriks eselon baris tereduksi, kita dapat menggunakan Operasi Baris Elementer (OBE) yang terdiri dari tiga jenis operasi, yaitu:

1. Forward phase: langkah untuk membuat nol di bawah leading one, sehingga membentuk matriks eselon baris.
2. Backward phase: langkah untuk membuat nol di atas leading one, sehingga membentuk matriks eselon baris tereduksi.

Setelah mendapatkan matriks eselon baris tereduksi dengan menggunakan Operasi Baris Elementer, kita dapat menulis beberapa persamaan yang sesuai dengan matriks tersebut. Dari persamaan-persamaan tersebut, kita dapat mencari solusi dari sistem persamaan linear secara cepat sebab semua variabel sudah berdiri sendiri sehingga kolom paling kanan selalu merupakan jawaban SPL.

2.3 Matriks kofaktor

Untuk suatu matriks A yang berukuran n x n, didefinisikan M_{ij} yang merupakan minor entri dan C_{ij} yang merupakan kofaktor entri. Minor entri (M_{ij}) didapat dengan menghitung determinan dari upamatriks (*submatrix*) yang elemen-elemennya tidak berada pada baris i dan kolom j. Sedangkan kofaktor entri dapat diperoleh melalui rumus berikut.

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

Gambar 2.3. Rumus kofaktor entri

Kofaktor entri berkoresponden dengan minor entri. Maka dari itu, jika sudah diketahui minor entri, kofaktor entri juga dapat diperoleh. Matriks kofaktor adalah kofaktor dari setiap elemen matriks A.

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

Gambar 2.4 Matriks Kofaktor

2.4 Determinan

Dalam konteks matriks persegi, determinan matriks merupakan suatu angka yang dapat dihubungkan dengan matriks tersebut. Matriks persegi adalah matriks yang memiliki jumlah baris dan kolom yang sama. Notasi determinan matriks A yang berukuran n x n adalah
|A| = det(A), yang dapat diilustrasikan sebagai berikut

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

Gambar 2.5 Determinan Matriks A

Terdapat 2 metode utama untuk mencari determinan matriks nxn, metode ekspansi kofaktor serta metode reduksi baris. Metode reduksi baris kita gunakan untuk mendapatkan bentuk matriks segitiga atas maupun segitiga bawah dengan melakukan Operasi Baris Elementer.

$$A = \begin{bmatrix} a_{11} & 0 & 0 & 0 \\ a_{12} & a_{22} & 0 & 0 \\ a_{13} & a_{23} & a_{33} & 0 \\ a_{14} & a_{24} & a_{34} & a_{44} \end{bmatrix}$$

Gambar 2.6 Matriks segitiga bawah

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ 0 & a_{22} & a_{23} & a_{24} \\ 0 & 0 & a_{33} & a_{34} \\ 0 & 0 & 0 & a_{44} \end{bmatrix}$$

Gambar 2.7 Matriks Segitiga

$$\det(A) = a_{11}a_{22}a_{33}a_{44}$$

$$\det(A) = a_{11}a_{22} \dots a_{nn}$$

Gambar 2.8 Rumus determinan Matriks Segitiga atas dan bawah

Determinan matriks segitiga atas dan bawah adalah hasil perkalian elemen diagonalnya Selain itu, terdapat beberapa sifat determinan yang berguna dalam berbagai manipulasi matriks, yaitu:

1. Jika suatu matriks A dikalikan dengan konstanta k pada salah satu barisnya sehingga menghasilkan matriks B, maka determinan matriks B adalah k kali determinan matriks A, yaitu det(B) = k * det(A).
2. Jika suatu matriks A mengalami pertukaran antara dua barisnya sehingga menghasilkan matriks B, maka determinan matriks B adalah negatif dari determinan matriks A, yaitu det(B) = -det(A).
3. Jika sebuah matriks A ditambah kelipatan k baris lainnya sehingga menghasilkan matriksB, maka det(B) = det(A).

Sifat-sifat tersebut adalah aturan penting yang digunakan dalam perhitungan determinan matriks dan manipulasi matriks agar menjadi matriks segitiga atas dan bawah. Hasil perkalian dari diagonal utama harus disesuaikan mengikuti aturan tersebut.

Cara lain untuk memperoleh determinan matriks adalah dengan menggunakan metode ekspansi kofaktor. Metode ini mengalikan elemen asli matriks dengan cofactornya pada satu baris ataupun kolom.

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

Gambar 2.9 Perhitungan determinan dengan ekspansi kofaktor secara baris

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

Gambar 2.10 Perhitungan dengan ekspansi kofaktor secara kolom

2.5 Matriks adjoin

Matriks adjoin merupakan transpose dari matriks kofaktor A. Proses transpose dilakukan dengan menurkan baris serta elemen suatu matriks. Baris 1 menjadi kolom 1 dan seterusnya.

2.6 Matriks balikan

Matriks balikan dari suatu matriks persegi A berukuran n x n didefinisikan sebagai matriks A^{-1} yang memenuhi syarat A * A^{-1} = A^{-1} * A = I, di mana I adalah matriks identitas n x n. Matriks identitas memiliki elemen 1 pada diagonal utamanya dan elemen 0 di tempat lainnya. Suatu matriks A memiliki matriks balikan jika dan hanya jika determinannya (det(A)) tidak sama dengan nol.

Ada dua metode utama untuk menghitung matriks balikan A: metode Eliminasi Gauss-Jordan dan metode menggunakan matriks adjoin.

Metode Eliminasi Gauss-Jordan digunakan untuk menghitung matriks balikan A. Untuk matriks A berukuran n x n, langkah-langkahnya adalah melakukan eliminasi pada matriks gabungan [A | I] hingga mendapatkan bentuk [I | A^{-1}], di mana I adalah matriks identitas n x n.

Metode kedua adalah dengan menggunakan matriks adjoin. Matriks balikan A, A^(-1), dapat dihitung dengan rumus yang melibatkan matriks adjoin, yaitu:

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

Gambar 2.11 Rumus dari matriks inverse

Kedua metode ini adalah cara yang umum digunakan untuk menghitung matriks balikan.

2.7 Kaidah Cramer

Kaidah Cramer adalah suatu metode yang digunakan untuk menentukan solusi dari sistem persamaan linear yang melibatkan matriks. Kaidah Cramer berfokus pada sistem persamaan linear yang memiliki jumlah persamaan sama dengan jumlah variabel dan memiliki matriks koefisien dengan determinan yang tidak sama dengan nol.

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

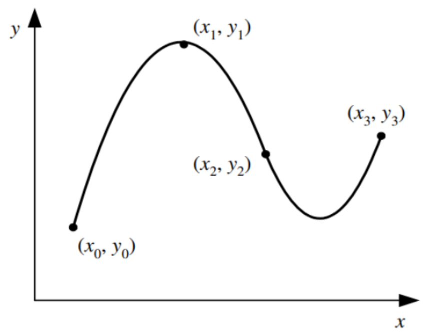
Gambar 2.12 Rumus kaidah cramer

Dalam hal ini, A_j adalah matriks yang diperoleh dengan mengganti entri dari kolom j pada matriks A dengan matriks b. Misal untuk mencari A_2 maka ganti kolom ke 2 matriks awal dengan matriks b yang merupakan matriks 1 kolom yang berisi hasil dari tiap SPL.

$$b = \begin{bmatrix} b_1 \\ b_2 \\ \ldots \\ b_n \end{bmatrix}$$

Gambar 2.13 Matriks b

2.8 Interpolasi polinom



Gambar 2.14 Ilustrasi interpolasi titik-titik

Interpolasi data merujuk pada pembuatan suatu kurva yang melewati sejumlah titik data tertentu. Ketika kurva tersebut berbentuk polinom, maka itu disebut sebagai polinom interpolasi. Polinom interpolasi dengan derajat n yang digunakan untuk menghubungkan titik-titik data $(x_0, y_0), (x_1, y_1), \dots, (x_n, y_n)$ berbentuk $p_n(x) = a_0 + a_1x + a_2x_2 + \cdots + a_nx_n$.

Jika hanya ada dua titik, (x_0, y_0) dan (x_1, y_1), maka polinom yang menginterpolasi kedua titik tersebut adalah $p_1(x) = a_0 + a_1x$ yang berupa persamaan garis lurus. Jika tersedia tiga titik, (x_0, y_0), (x_1, y_1) dan (x_2, y_2), maka polinom yang menginterpolasi ketiga titik tersebut adalah $p_2(x) = a_0 + a_1x + a_2x_2$, demikian seterusnya.

Dengan cara yang sama, kita dapat memeroleh polinom interpolasi berderajat n untuk n yang lebih tinggi asalkan tersedia (n+1) buah titik data. Dengan melakukan substitusi (xi, yi) ke dalam persamaan polinom $pn(x) = a_0 + a_1x + a_2x^2 + \cdots + a_nx^n$ untuk $i = 0, 1, 2, \dots, n$ akan diperoleh n buah sistem persamaan lanjar dalam $a_0, a_1, a_2, \dots, a_n$.

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

Gambar 2.15. Sistem persamaan lanjar hasil substitusi polinom

Solusi sistem persamaan lanjar ini, yaitu nilai $a_0, a_1, a_2, \dots, a_n$ dapat diperoleh dengan menggunakan metode eliminasi Gauss. Sebagai contoh, diberikan tiga buah titik yaitu (8.0, 2.0794), (9.0, 2.1972), dan (9.5, 2.2513). Tentukan polinom interpolasi kuadratik lalu estimasi nilai fungsi pada x = 9.2. Polinom kuadratik berbentuk $p2(x) = a0 + a1x + a2x2$. Dengan melakukan substitusi ketiga buah titik data ke dalam polinom tersebut, diperoleh sisten persamaan lanjar sebagai berikut

$$a_0 + 8.0a_1 + 64.00a_2 = 2.0794$$
$$a_0 + 9.0a_1 + 81.00a_2 = 2.1972$$
$$a_0 + 9.5a_1 + 90.25a_2 = 2.2513$$

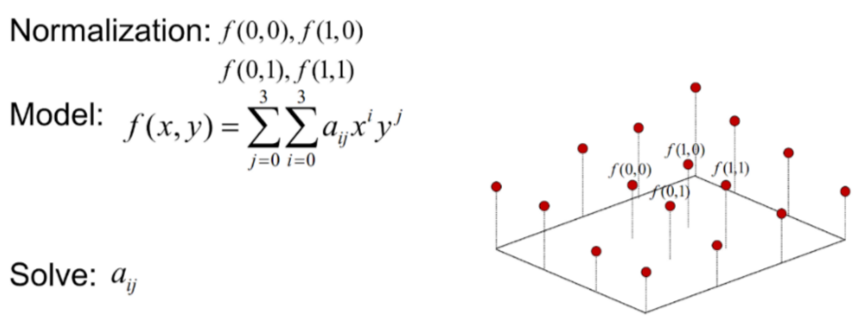
Gambar 2.16 Sistem Persamaan Lanjar hasil substitusi titik

Penyelesaian sistem persamaan linear tersebut dengan metode eliminasi Gauss menghasilkan a_0 = 0.6762, a_1 = 0.2266, a_2 = -0.0064. Polinom interpolasi yang melalui ketiga buah titik tersebut adalah $p_2(x)$ = 0.6762 + 0.2266x - 0.0064x^2. Dengan menggunakan polinom ini, maka nilai fungsi pada x = 9.2 dapat ditaksir sebagai berikut: $p_2(9.2) = 0.6762 + 0.2266(9.2) - 0.0064(9.2)^2 = 2.2192$.

2.9 Interpolasi *bicubic spline*

Interpolasi bicubic spline adalah metode yang digunakan untuk mendekati sebuah fungsi di antara titik-titik data yang sudah dikenal. Metode ini melibatkan penggunaan spline dan pembangunan sejumlah polinomial kubik di dalam setiap sel segi empat dari data yang ada. Pendekatan ini menghasilkan permukaan yang halus dan kontinu, yang memungkinkan visualisasi data yang lebih akurat daripada menggunakan metode interpolasi linear.

Dalam proses interpolasi bicubic spline, kita menggunakan 16 titik, dengan 4 titik utama di pusat dan 12 titik lainnya di sekitarnya. Titik-titik ini digunakan untuk mendekati turunan dari keempat titik utama dan membangun permukaan bicubic spline. Model ini membantu dalam memahami dan menggambarkan hubungan antara data secara lebih halus dan detail.



Gambar 2.17 Permodelan masalah Interpolasi *bicubic spline*

Selain menggunakan model dasar, juga diperlukan model turunan yang memiliki arah dari kedua sumbu, baik terhadap sumbu x, sumbu y, atau keduanya. Persamaan polinomial yang diterapkan adalah seperti yang berikut.

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

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

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

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

Gambar 2.18 Persamaan polinomial yang digunakan

$$y = Xa$$

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

Gambar 2.19 Matriks hasil persamaan polinomial yang digunakan

$$a = X^{-1}DI$$

$$\begin{bmatrix} a_{00} \\ a_{10} \\ a_{20} \\ a_{30} \\ a_{01} \\ a_{11} \\ a_{21} \\ a_{31} \\ a_{02} \\ a_{12} \\ a_{22} \\ a_{32} \\ a_{03} \\ a_{13} \\ a_{23} \\ a_{33} \end{bmatrix} = \frac{1}{16} \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 16 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -8 & 0 & 8 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 16 & -40 & 32 & -8 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -8 & 24 & -24 & 8 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -8 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 8 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -4 & 0 & 0 & -4 & 4 & 0 & 0 & 0 & 0 & 4 & 0 & 0 & 0 & 0 & 0 \\ 0 & 32 & -20 & 0 & 8 & -4 & -4 & 0 & 0 & -24 & 16 & -4 & 0 & 0 & 0 & 0 \\ 0 & -20 & 12 & 0 & -4 & 0 & 4 & 0 & 0 & 16 & -12 & 4 & 0 & 0 & 0 & 0 \\ 0 & 16 & 0 & 0 & 0 & -40 & 0 & 0 & 0 & 32 & 0 & 0 & 0 & -8 & 0 & 0 \\ 0 & 8 & 0 & 0 & 32 & -4 & -24 & 0 & -20 & -4 & 0 & 0 & 0 & 0 & -4 & 0 \\ 0 & -64 & 40 & 0 & -64 & 96 & -68 & 24 & 40 & -68 & -16 & -16 & 0 & 24 & -16 & 0 \\ 0 & 40 & -24 & 0 & 32 & -52 & 52 & -24 & -20 & 40 & 16 & 16 & 0 & -16 & 12 & 0 \\ 0 & -8 & 0 & 0 & 0 & 24 & 0 & 0 & 0 & -24 & 0 & 0 & 0 & 8 & 0 & 4 \\ 0 & -4 & 0 & 0 & -20 & 0 & 16 & 0 & 12 & 4 & -12 & 0 & 0 & 0 & 4 & -4 \\ 0 & 32 & -20 & 0 & 40 & -52 & 40 & -16 & -24 & 52 & -52 & 12 & 0 & -24 & 16 & -4 \\ 0 & -20 & 12 & 0 & -20 & 28 & -32 & 16 & 12 & -32 & 40 & -12 & 0 & 16 & -12 & 4 \end{bmatrix} \begin{bmatrix} I(-1,-1) \\ I(0,-1) \\ I(1,-1) \\ I(2,-1) \\ I(-1,0) \\ I(0,0) \\ I(1,0) \\ I(2,0) \\ I(-1,1) \\ I(0,1) \\ I(1,1) \\ I(2,1) \\ I(-1,2) \\ I(0,2) \\ I(1,2) \\ I(2,2) \end{bmatrix}$$

Gambar 2.20 Matriks persamaan polinomial untuk modifikasi gambar

Elemen-elemen dalam matriks X adalah representasi dari setiap koefisien a_{ij} yang berasal dari persamaan fungsi dan persamaan turunan yang telah dijelaskan sebelumnya. Sebagai contoh, elemen matriks X di baris ke-8 dan kolom ke-2 mewakili koefisien a_{10} dalam ekspansi sigma untuk $f_x(1, 1)$, yang menghasilkan nilai konstanta $1 \times 1^{1-1} \times 1^0 = 1$, sesuai dengan nilai yang terdapat dalam matriks X.

Nilai dari vektor a dapat dihitung dari persamaan y = Xa, dan vektor a tersebut kemudian digunakan sebagai variabel dalam f(x, y), yang pada akhirnya membentuk fungsi interpolasi bicubic sesuai dengan model yang telah dijelaskan.

2.10 Regresi linear berganda

Regresi Linear adalah salah satu teknik yang digunakan untuk memprediksi nilai selain dari metode Interpolasi Polinom. Meskipun terdapat persamaan khusus untuk menghitung regresi linear sederhana, ada juga persamaan umum yang digunakan dalam regresi linear berganda.

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

Gambar 2.20 Persamaan umum regresi linear

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

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

Gambar 2.21 *Normal Estimation Equationfor Multiple Linear Regression*

Sistem persamaan linear tersebut diselesaikan dengan menggunakan metode eliminasi Gauss.

BAB III

IMPLEMENTASI PUSTAKA DAN PROGRAM

3.1 Struktur Code Program

Source code secara umum dibagi menjadi 4 struktur dasar utama yang disimpan dalam src folder, yaitu :

1. Aplikasi : Berisi menu dari masing-masing program yang akan dipanggil di main
2. Matrix : Berisi abstraksi tipe data Matrix, serta implementasi dari pustaka-pustaka yang akan digunakan di aplikasi
3. Utils : Berisi beberapa function yang umum digunakan di program-program lain
4. Main : Program utama untuk menggabungkan berbagai aplikasi yang kami miliki

3.2 Pustaka Matrix

3.2.1 Atribut

Tabel 1. Atribut dari *Class* Matrix

Nama Atribut	Penjelasan Atribut
public int row	Mendeklarasikan jumlah baris matriks
public int col	Mendeklarasikan jumlah kolom matriks
public double [][] matrix	Mendeklarasikan besar *array of array* inisialisasi matriks

3.2.2 Metode

Tabel 2. Metode dari *Class* Matrix

Nama Metode	Penjelasan Metode
public double getElmt (**int** i, **int** j)	Mendapatkan elemen Matrix [i][j]
Public void readMatrixCLI (**int** row, **int** col)	Prosedur untuk melakukan *input* matriks ke sebuah matrix yang sudah dideklarasikan melalui keyboard dengan jumlah baris = row,

	jumlah kolom = col.
public void readMatrixFile (**String** filename)	Prosedur untuk melakukan pembacaan matrix dari sebuah file yang diinput oleh user dengan nama file = filename.
public void printMatrix()	Prosedur untuk melakukan pencetakan matriks ke terminal.
public void roundElmtMatrix (**int** n)	Prosedur untuk melakukan *rounding* dari setiap elemen matriks hingga scale yang kita ingin gunakan.
public Matrix multiplyMatrix (**Matrix** m1, **Matrix** m2)	Prosedur untuk melakukan perkalian antara Matrix m1 dan Matrix m2.

3.3 Pustaka SPL

3.3.1 Atribut

Tidak terdapat Atribut pada Class ini.

3.3.2 Metode

Tabel 3. Metode dari *Class* SPL

Nama Metode	Penjelasan Metode
public static int findIndexColFirstNonZero (**Matrix** matrix, **int** row)	Function untuk mencari indeks koloom pertama kemunculan angkan non-0 di suatu baris. Mengembalikan -1 jika tidak ada.
public static void createFirstOne (**Matri**x matrix, **int** row, **int** colTambahan)	Membuat angka 1 pertama di setiap barisnya. colTambahan adalah kolom yang ditambahkan pada matriks normal, misal pada persamaan, ditambah 1 kolom tambahan, lalu pada invers atau determinan, tidak ada kolom tambahan.
public static void divideRowSelf (**Matrix** matrix, **int** row, **double** divider)	Membagi Matriks = matrix pada baris = row dengan pembagi sejumlah = divider. Digunakan untuk membagi
public static void kaliRow	Prosedur untuk melakukan perkalian di setiap

(**Matrix** matrix, **double** pengali, **int** row)	elemen baris = row pada matriks = matrix sejumlah pengali.
public static Matrix RetKaliRow (**Matrix** matrix, **double** pengali, **int** row)	Fungsi yang mengembalikan matrix yang berisi hasil perkalian suatu baris = row pada Matriks = matrix sejumlah pengali.
public static void bagiRow (**Matrix** matrix, **double** pengali, **int** row)	Prosedur untuk melakukan perkalian antara Matrix m1 dan Matrix m2.
public static void kurangRow (**Matrix** matrix, **int** rowAsal, **int** rowPengurang)	Prosedur untuk mengurangi satu row dengan ro lainnya.
public static void swapRow (**Matrix** matrix, **int** rowAsal, **int** rowTujuan)	Prosder untuk menukar satu baris dengan baris lainnya.
public static int checkSolveType (**Matrix** matrix, **int** colTambahan)	Prosedur untuk menentukan tipe *solve* dari suatu matriks eselon baris.
public static void CreateMatrixEselon (**Matrix** matrix, **int** colTambahan)	Prosedur untuk membuat matriks eselon baris dari suatu matriks.
public static void CreateMatrixEselonReduced (**Matrix** matrix, **int** coltambahan)	Prosedur untuk membuat matriks eselon baris tereduksi dari suatu matriks eselon baris.
public static int nZero (**Matrix** matrix, **int** row, **int** colTambahan)	Fungsi untuk menghitung jumlah 0 pada suatu baris.
public static void CreateReducedParametric (**Matrix** matrix)	Prosedur untuk membuat suatu matriks eselon baris tereduksi apabila solusinya adalah parametric.
public static void parametricSol (**Matrix** matrix)	Prosedur untuk menyelesaikan Matrix eselon baris yang solusinya adalah parametrik.
public static void solveSPLEchelon (**Matrix** matrix, **int** colTambahan)	Prosedur untuk menyelesaikan Matrix eselon baris yang solusinya bukan parametrik serta tidak ada.
public static void solveSPLReduced (**Matrix** matrix)	Prosedur untuk menyelesaikan Matrix eselon baris tereduksi.
public static void recursionSolve (**Matrix** matrix, **Matrix** solution, **int** row)	Prosedur untuk membantu menyelesaikan Matrix eselon baris secara rekursif
public static Matrix getEquationOnly	Prosedur untuk mendapatkan konstanta dari

(**Matrix** m)	peubah pada SPL saja, tanpa mempedulikan hasilnya. (Untuk Cramer).
public static Matrix getAnsOnly (**Matrix** m)	Prosedur untuk mendapatkan hasil dari SPL saja, tanpa mempedulikan konstanta dari peubah. (Untuk Cramer).
public static void CramerSolver (**Matrix** m, **String[]** s)	Prosedur untuk menyelesaikan SPL dengan metode Cramer.
public static void inverseSPL (**double[][]** mtrx, **double[]** constant)	Prosedur untuk menyelesaikan SPL dengan metode matriks balikan.

3.4 Pustaka Inverse

3.4.1 Atribut

Tidak terdapat Atribut pada Class ini.

3.4.2 Metode

Tabel 4. Metode dari *Class* Inverse

Nama Metode	Penjelasan Metode
static double[][] getCofactor (**double[][]** A, **int** given_row, **int** given_column)	Fungsi yang akan mengembalikan Cofactor dari row dan column input.
static double determinant(**double[][]** A)	Fungsi untuk mencari determinan dari input
static void adjoint (**double[][]** A, **double [][]**adj)	Fungsi untuk mencari adjoint dari sebuah matriks.
static boolean inverse(**double[][]** A, **double [][]**inverse)	Fungsi untuk mencari inverse dari suatru matriks input.
static void inverseGaussJordan (**Matrix** matrix)	Prosedur untuk menghitung hasil SPL dengan metode matriks balikan.

3.5 Pustaka Determinan

3.5.1 Atribut

Tidak terdapat Atribut pada Class ini.

3.5.2 Metode

Tabel 5. Metode dari *Class* Determinan

Nama Metode	Penjelasan Metode
public static Matrix submatrix (**Matrix** matrix, **int** idxbaris, **int** idxkolom)	Fungsi untuk mengembalikan *submatrix* dari sebuah matriks di baris dan kolom tertentu.
public static double det2x2(**Matrix** matrix)	Fungsi untuk mencari determinan dari determinan khusus 2 x 2;
public static double detKof(**Matrix** matrix)	Fungsi untuk menghitung determinan dengan metode ekspansi kofaktor secara rekursif.
public static int getFirstNonZeroRowIdx (**Matrix** matrix, **int** idxcol)	Fungsi untuk mendapatkan index tidak 0 pertama pada suatu baris
public static double detReduksi (**Matrix** matrix)	Fungsi untuk mendapatkan determinan dengan metode reduiksi baris.

3.6 Program SPLApp

3.6.1 Penjelasan program

SPLApp merupakan program yang memanggil pustaka-pustaka untuk menyelesaikan permasalahan Sistem Persamaan Linear.

3.6.2 Metode

Tabel 6. Metode dari Program SPLApp

Nama Metode	Penjelasan Metode
public static void Menu()	Prosedur untuk menyelesaikan SPL, dengan 4 metode utama yaitu Metode Eliminasi Gauss, Metode Eliminasi Gauss Jordan, Metode

	Matriks Balikan, serta Metode Kaidah Cramer.

3.7 Program RLB

3.7.1 Penjelasan program

RLB merupakan program untuk menyelesaikan permasalahan Regresi Linear Berganda dengan input beberapa titik yang ada, serta menaksirkan nilai dari titik yang ingin ditaksir dari hasil regresi tersebut.

3.7.2 Metode

Tabel 7. Metode dari Program RLB

Nama Metode	Penjelasan Metode
public static void inputRLBFile (**Matrix** titik, **Matrix** ans)	Prosedur untuk input data yang dibutuhkan pada App RLB melalui file. Titik akan disimpan di matriks titik dan titik taksiran disimpan di matriks ans.
public static void inputRLBkey (**Matrix** titik, **Matrix** ans)	Prosedur untuk input data yang dibutuhkan pada App RLB melalui keyboard. Titik akan disimpan di matriks titik dan titik taksiran disimpan di matriks ans.
public static Matrix normaleq(**Matrix** m)	Membuat matrix normal equation dari titik-titik yang ada
public static void output (**Matrix** eselon, **Matrix** ans, **String[]** s)	Menghasilkan output dari fungsi RLB yang terbentuk.
public static void menu()	Menu utama dari program, tempat user berinteraksi dengan fungsi -fungsi yang telah dibuat

3.8 Program InterpolinomApp

3.8.1 Penjelasan program

Interpolinom merupakan program yang berisi kalkulator untuk mencari taksiran dari suatu nilai menggunakan metode interpolasi polinomial.

3.8.2 Metode

Tabel 8. Metode dari Program RLB

Nama Metode	Penjelasan Metode
public static void makeEq (**Matrix** matrix)	Membuat persamaan yang akan di *solve* untuk mendapatkan persamaan utamanya.
public static void solveEq(**Matrix** matrix)	Prosedur untuk menyelesaikan persamaan yang didapatkan.
public static Matrix saveEqInArr (**Matrix** matrix)	Menyimpan konstanta hasil persamaan dalam sebuah array.
public static double approxValue (**Matrix** eq, **double** x)	Mencari nilai aproksimasi dari fungsi yang terbentuk.
public static void Menu()	Menu utama program yang berinteraksi dengan pengguna. Dilakukan input serta output disini.

3.9 Program DeterminanApp

3.9.1 Penjelasan program

Program untuk menghitung determinan dengan metode kofaktor ataupun reduksi baris.

3.9.2 Metode

Tabel 9. Metode dari Program RLB

Nama Metode	Penjelasan Metode
public static void menu()	Berisi *interface* utama dari program yang memanggil library pencari determinan dari sebuah matriks. Metode yang digunakan adalah kofaktor dan reduksi baris.

3.10 Program InverseApp

3.10.1 Penjelasan program

Program ini mencari inverse dari suatu matriks dengan menggunakan metode adjoin ataupun matriks balikan.

3.10.2 Metode

Tabel 10. Metode dari Program InverseApp

Nama Metode	Penjelasan Metode
public static void menu()	Program utama untuk berinteraksi dengan pengguna. Berisi *input output* serta pemanggilan pustaka-pustaka yang dibutuhkan untuk menghitung inverse dari suatu Matriks.

3.11 Program BicubicSpline

3.11.1 Penjelasan program

Program ini melakukan aproksimasi atas suatu nilai titik yang diberikan pengguna pada range tertentu dengan menggunakan metode *bicubic spline interpolation*.

3.11.2 Metode

Tabel 11. Metode dari Program BicubicSpline

Nama Metode	Penjelasan Metode
public static void menu()	Program utama untuk berinteraksi dengan pengguna. Berisi *input output* serta pemanggilan pustaka-pustaka yang dibutuhkan untuk menghitung inverse dari suatu Matriks.
static double[][] InverseOfMatrix (**double[][]** matrix, **int** order)	Mencari inverse dari matrix dengan size order
static double[][] matrixX (**double[][]** A)	Membentuk matrix X untuk bicubic spline interpolation
static double BicubicSplineEquation(double[][] variables, double a, double b)	Melakukan value modifikasi sesuai model *bicubic spline* yang menjadi rumus
static double[][] InverseOfMatrix(double[][] matrix, int order	Mencari inverse dari matrix dengan size order

3.12 Program BonusSplineImage

3.12.1 Penjelasan program

Program ini melakukan perbesaran gambar dengan suatu skala tertentu, lalu diperhalus menggunakan konstanta a yang didapat dari interpolasi *bicubic spline*.

3.12.2 Metode

Tabel 12. Metode dari Program BonusSplineImage

Nama Metode	Penjelasan Metode
public static int[][] add_border(**int A[][]**)	Menambahkan baris terluar dari matrix dengan melakukan copy elemen-elemen terluar (indeks row dan kolom sebesar 0 atau setara length-1) agar bisa diolah dengan algoritma *bicubic spline interpolation* yang

	membutuhkan acuan titik sekitar.
public static byte[] iterate_image (**int** A[][], **int** scale)	Mengolah array of bytes dari image yang didapatkan dengan mengambil normalisasi titik serta memecah tiap byte menjadi alpha, red, green, blue bytes serta memanipulasi valuenya dengan matrix X^-1*D dengan *metode bicubic spline interpolation*
static double[][] matrixX (**double[][]** A)	Membentuk matrix X untuk bicubic spline interpolation
static double[][] matrixD (double A[][])	Membentuk matrix D untuk manipulasi pixel gambar
public static BufferedImage resize (**String** inputImagePath, **String** outputImagePath, **int** scaledWidth, **int** scaledHeight)	Resize image dan menghasilkan image sesuai spesifikasi scaledWidth dan scaledHeight ke outputImagePath
Public static BufferedImage resize_scale(**String** inputImagePath, **String** outputImagePath, **int** scale)	Melakukan perhitungan width dan height gambar sesuai scale lalu ditembuskan ke resize_scale
private static int[][] convertTo2DWithoutUsingGetRGB(BufferedImage image)	Mengubah image yang menjadi input menjadi sebuah matrix 2 dimensi
static double BicubicSplineEquation(double[][] variables, double a, double b)	Melakukan value modifikasi dari pixel gambar
static double[][] InverseOfMatrix(double[][] matrix, int order	Mencari inverse dari matrix dengan size order
static double[][] multiplyMatrix(int row1, int col1, double A[][], int row2, int col2, double B[][])	Melakukan perkalian matrix A berukuran row 1 x col1 dengan matrix B berukuran row2 x col2 dengan syarat row1 = col2
static double[][] inverseSPLBicubicSpline(double inversed[][], int constant[])	Mencari nilai a dengan SPL menggunakan matrix sesuai rumus

BAB IV
EKSPERIMEN

4.1 Percobaan Fungsi Sistem Persamaan Linear

1a.

<table>
<tr><th>Input</th><th>Output</th></tr>
<tr><td rowspan="8">$$A = \begin{bmatrix} 1 & 1 & -1 & -1 \\ 2 & 5 & -7 & -5 \\ 2 & -1 & 1 & 3 \\ 5 & 2 & -4 & 2 \end{bmatrix}, \quad b = \begin{bmatrix} 1 \\ -2 \\ 4 \\ 6 \end{bmatrix}$$</td><td>METODE GAUSS</td></tr>
<tr><td>==========SPL==========
1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Metode matriks balikan
4. Metode Cramer
Masukkan jenis penyelesaian SPL: 1
Masukkan matrix melalui terminal (1) atau file (2)?: 2
Masukkan path file (relatif terhadap folder test): 1_a.txt
Tidak ada penyelesaian untuk matriks ini.</td></tr>
<tr><td>ELIMINASI GAUSS JORDAN</td></tr>
<tr><td>==========SPL==========
1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Metode matriks balikan
4. Metode Cramer
Masukkan jenis penyelesaian SPL: 2
Masukkan matrix melalui terminal (1) atau file (2)?: 2
Masukkan path file (relatif terhadap folder test): 1_a.txt
Tidak ada solusi yang memenuhi.</td></tr>
<tr><td>MATRIKS BALIKAN</td></tr>
<tr><td>==========SPL==========
1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Metode matriks balikan
4. Metode Cramer
Masukkan jenis penyelesaian SPL: 3
Masukkan matrix melalui terminal (1) atau file (2)?: 2
Masukkan path file (relatif terhadap folder test): 1_a.txt
Matriks tidak memiliki invers!</td></tr>
<tr><td>KAIDAH CRAMER</td></tr>
<tr><td>==========SPL==========
1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Metode matriks balikan
4. Metode Cramer
Masukkan jenis penyelesaian SPL: 4
Masukkan matrix melalui terminal (1) atau file (2)?: 2
Masukkan path file (relatif terhadap folder test): 1_a.txt
Determinan == 0, maka solusi dari SPL di atas tidak unik, gunakan metode lain.</td></tr>
</table>

1b.

<table>
<tr><th>Input</th><th>Output</th></tr>
<tr><td rowspan="8">$$A = \begin{bmatrix} 1 & -1 & 0 & 0 & 1 \\ 1 & 1 & 0 & -3 & 0 \\ 2 & -1 & 0 & 1 & -1 \\ -1 & 2 & 0 & -2 & -1 \end{bmatrix}, \quad b = \begin{bmatrix} 3 \\ 6 \\ 5 \\ -1 \end{bmatrix}$$</td><td>METODE GAUSS</td></tr>
<tr><td>==========SPL==========
1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Metode matriks balikan
4. Metode Cramer
Masukkan jenis penyelesaian SPL: 1
Masukkan matrix melalui terminal (1) atau file (2)?: 2
Masukkan path file (relatif terhadap folder test): spl1b.txt
Solusi akhir:
X1 = 3.6666666666666665
X2 = 1.333333333333334
X3 = t1
X4 = -0.33333333333333315
X5 = 0.6666666666666666</td></tr>
<tr><td>ELIMINASI GAUSS JORDAN</td></tr>
<tr><td>==========SPL==========
1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Metode matriks balikan
4. Metode Cramer
Masukkan jenis penyelesaian SPL: 2
Masukkan matrix melalui terminal (1) atau file (2)?: 2
Masukkan path file (relatif terhadap folder test): spl1b.txt
Solusi akhir:
X1 = 3.6666666666666665
X2 = 1.333333333333334
X3 = t1
X4 = -0.33333333333333315
X5 = 0.6666666666666666</td></tr>
<tr><td>MATRIKS BALIKAN</td></tr>
<tr><td>==========SPL==========
1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Metode matriks balikan
4. Metode Cramer
Masukkan jenis penyelesaian SPL: 3
Masukkan matrix melalui terminal (1) atau file (2)?: 2
Masukkan path file (relatif terhadap folder test): spl1b.txt
Matriks tidak memiliki invers!</td></tr>
<tr><td>KAIDAH CRAMER</td></tr>
<tr><td>==========SPL==========
1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Metode matriks balikan
4. Metode Cramer
Masukkan jenis penyelesaian SPL: 4
Masukkan matrix melalui terminal (1) atau file (2)?: spl1b.txt
Input tidak valid!
Masukkan matrix melalui terminal (1) atau file (2)?: 2
Masukkan path file (relatif terhadap folder test): spl1b.txt
Determinan == 0, maka solusi dari SPL di atas tidak unik, gunakan metode lain.</td></tr>
</table>

1c.

Input	Output

<table>
<tr><td rowspan="8">$$A = \begin{bmatrix} 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}, \quad b = \begin{bmatrix} 2 \\ -1 \\ 1 \end{bmatrix}$$</td><td>METODE GAUSS</td></tr>
<tr><td>==========SPL==========
1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Metode matriks balikan
4. Metode Cramer
Masukkan jenis penyelesaian SPL: 1
Masukkan matrix melalui terminal (1) atau file (2)?: 2
Masukkan path file (relatif terhadap folder test): spl1c.txt
Solusi akhir:
X1 = t2
X2 = (1.0 - t1)
X3 = t2
X4 = (-2.0 - t1)
X5 = (1.0 - -1.0*t1)
X6 = t1</td></tr>
<tr><td>ELIMINASI GAUSS JORDAN</td></tr>
<tr><td>==========SPL==========
1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Metode matriks balikan
4. Metode Cramer
Masukkan jenis penyelesaian SPL: 2
Masukkan matrix melalui terminal (1) atau file (2)?: 2
Masukkan path file (relatif terhadap folder test): spl1c.txt
Solusi akhir:
X1 = t2
X2 = (1.0 - t1)
X3 = t2
X4 = (-2.0 - t1)
X5 = (1.0 - -1.0*t1)
X6 = t1</td></tr>
<tr><td>MATRIKS BALIKAN</td></tr>
<tr><td>==========SPL==========
1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Metode matriks balikan
4. Metode Cramer
Masukkan jenis penyelesaian SPL: 3
Masukkan matrix melalui terminal (1) atau file (2)?: 2
Masukkan path file (relatif terhadap folder test): spl1c.txt
Matriks tidak memiliki invers!</td></tr>
<tr><td>KAIDAH CRAMER</td></tr>
<tr><td>==========SPL==========
1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Metode matriks balikan
4. Metode Cramer
Masukkan jenis penyelesaian SPL: 4
Masukkan matrix melalui terminal (1) atau file (2)?: 2
Masukkan path file (relatif terhadap folder test): spl1c.txt
Determinan == 0, maka solusi dari SPL di atas tidak unik, gunakan metode lain.</td></tr>
</table>

1d.

N = 6

Input	Output

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

METODE GAUSS

```
==========SPL==========
1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Metode matriks balikan
4. Metode Cramer
Masukkan jenis penyelesaian SPL: 1
Masukkan matrix melalui terminal (1) atau file (2)?: 2
Masukkan path file (relatif terhadap folder test): hilbert.txt
Solusi persamaan adalah:
X0 = 35.999910
X1 = -629.998656
X2 = 3359.997943
X3 = -7560.009266
X4 = 7560.023193
X5 = -2772.013311
```

ELIMINASI GAUSS JORDAN

```
==========SPL==========
1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Metode matriks balikan
4. Metode Cramer
Masukkan jenis penyelesaian SPL: 2
Masukkan matrix melalui terminal (1) atau file (2)?: 2
Masukkan path file (relatif terhadap folder test): hilbert.txt
Nilai setiap variabel adalah:
X1 = 35.999910
X2 = -629.998656
X3 = 3359.997943
X4 = -7560.009266
X5 = 7560.023193
X6 = -2772.013311
```

MATRIKS BALIKAN

```
==========SPL==========
1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Metode matriks balikan
4. Metode Cramer
Masukkan jenis penyelesaian SPL: 3
Masukkan matrix melalui terminal (1) atau file (2)?: 2
Masukkan path file (relatif terhadap folder test): hilbert.txt
Solusi persamaan:
X1 = 35.999998
X2 = -629.999950
X3 = 3359.999710
X4 = -7559.999411
X5 = 7559.999516
X6 = -2771.999864
```

KAIDAH CRAMER

```
==========SPL==========
1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Metode matriks balikan
4. Metode Cramer
Masukkan jenis penyelesaian SPL: 4
Masukkan matrix melalui terminal (1) atau file (2)?: 2
Masukkan path file (relatif terhadap folder test): hilbert.txt
Solusi persamaan adalah:
x1 = 36.0000
x2 = -630.0000
x3 = 3360.0000
x4 = -7560.0000
x5 = 7560.0000
x6 = -2772.0000
```

N = 10

<table>
<tr><th>Input</th><th>Output</th></tr>
<tr><td rowspan="7">

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

</td><td>METODE GAUSS</td></tr>
<tr><td>

```
==========SPL==========
1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Metode matriks balikan
4. Metode Cramer
Masukkan jenis penyelesaian SPL: 1
Masukkan matrix melalui terminal (1) atau file (2)?: 2
Masukkan path file (relatif terhadap folder test): hilbert2.txt
Solusi persamaan adalah:
X0 = 47.285443
X1 = -1011.823653
X2 = 5318.458922
X3 = -813.818260
X4 = -59760.320648
X5 = 156659.906509
X6 = -113485.771767
X7 = -84055.121476
X8 = 156829.193851
X9 = -59739.816688
```

</td></tr>
<tr><td>ELIMINASI GAUSS JORDAN</td></tr>
<tr><td>

```
==========SPL==========
1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Metode matriks balikan
4. Metode Cramer
Masukkan jenis penyelesaian SPL: 2
Masukkan matrix melalui terminal (1) atau file (2)?: 2
Masukkan path file (relatif terhadap folder test): hilbert2.txt
Nilai setiap variabel adalah:
X1 = 47.285443
X2 = -1011.823653
X3 = 5318.458922
X4 = -813.818260
X5 = -59760.320648
X6 = 156659.906509
X7 = -113485.771767
X8 = -84055.121476
X9 = 156829.193851
X10 = -59739.816688
```

</td></tr>
<tr><td>MATRIKS BALIKAN</td></tr>
<tr><td>

```
==========SPL==========
1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Metode matriks balikan
4. Metode Cramer
Masukkan jenis penyelesaian SPL: 3
Masukkan matrix melalui terminal (1) atau file (2)?: 2
Masukkan path file (relatif terhadap folder test): hilbert2.txt
Solusi persamaan:
X1 = 34.590178
X2 = -381.540248
X3 = 0.375622
X4 = 5698.242961
X5 = -9412.043303
X6 = -3176.675595
X7 = 7577.607165
X8 = 5393.857417
X9 = -2806.710061
X10 = -3005.422213
```

</td></tr>
<tr><td>KAIDAH CRAMER</td></tr>
</table>

	==========SPL========== 1. Metode eliminasi Gauss 2. Metode eliminasi Gauss-Jordan 3. Metode matriks balikan 4. Metode Cramer Masukkan jenis penyelesaian SPL: 4 Masukkan matrix melalui terminal (1) atau file (2)?: 2 Masukkan path file (relatif terhadap folder test): hilbert2.txt Solusi persamaan adalah: x1 = 99.9951 x2 = -4949.7685 x3 = 79193.9767 x4 = -600545.3797 x5 = 2522259.5072 x6 = -6305582.8483 x7 = 9608421.2569 x8 = -8750459.1754 x9 = 4375200.3205 x10 = -923647.9793

2a.

Input	Output
$$\begin{bmatrix} 1 & -1 & 2 & -1 & -1 \\ 2 & 1 & -2 & -2 & -2 \\ -1 & 2 & -4 & 1 & 1 \\ 3 & 0 & 0 & -3 & -3 \end{bmatrix}.$$	METODE GAUSS
	==========SPL========== 1. Metode eliminasi Gauss 2. Metode eliminasi Gauss-Jordan 3. Metode matriks balikan 4. Metode Cramer Masukkan jenis penyelesaian SPL: 1 Masukkan matrix melalui terminal (1) atau file (2)?: 2 Masukkan path file (relatif terhadap folder test): 2_a.txt Solusi akhir: X1 = (-1.0 - -1.0*t2) X2 = (0.0 - -2.0*t1) X3 = t1 X4 = t2
	ELIMINASI GAUSS JORDAN

```
==========SPL==========
1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Metode matriks balikan
4. Metode Cramer
Masukkan jenis penyelesaian SPL: 2
Masukkan matrix melalui terminal (1) atau file (2)?: 2
Masukkan path file (relatif terhadap folder test): 2_a.txt
Solusi akhir:
X1 = (-1.0 - -1.0*t2)
X2 = (0.0 - -2.0*t1)
X3 = t1
X4 = t2
```

MATRIKS BALIKAN

```
==========SPL==========
1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Metode matriks balikan
4. Metode Cramer
Masukkan jenis penyelesaian SPL: 3
Masukkan matrix melalui terminal (1) atau file (2)?: 2
Masukkan path file (relatif terhadap folder test): 2_a.txt
Matriks tidak memiliki invers!
```

KAIDAH CRAMER

```
==========SPL==========
1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Metode matriks balikan
4. Metode Cramer
Masukkan jenis penyelesaian SPL: 4
Masukkan matrix melalui terminal (1) atau file (2)?: 2
Masukkan path file (relatif terhadap folder test): 2_a.txt
Determinan == 0, maka solusi dari SPL di atas tidak unik, gunakan metode lain.
```

2.b

Input	Output
$\begin{bmatrix} 1 & -1 & 2 & -1 & -1 \\ 2 & 1 & -2 & -2 & -2 \\ -1 & 2 & -4 & 1 & 1 \\ 3 & 0 & 0 & -3 & -3 \end{bmatrix}.$	==========SPL========== 1. Metode eliminasi Gauss 2. Metode eliminasi Gauss-Jordan 3. Metode matriks balikan 4. Metode Cramer Masukkan jenis penyelesaian SPL: 4 Masukkan matrix melalui terminal (1) atau file (2)?: 2 Masukkan path file (relatif terhadap folder test): 2_a.txt Determinan == 0, maka solusi dari SPL di atas tidak unik, gunakan metode lain.

3.a

Input	Output

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

METODE GAUSS

```
==========SPL==========
1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Metode matriks balikan
4. Metode Cramer
Masukkan jenis penyelesaian SPL: 1
Masukkan matrix melalui terminal (1) atau file (2)
?: 2
Masukkan path file (relatif terhadap folder test):
 3_a.txt
Solusi persamaan adalah:
X0 = -0.224324
X1 = 0.182432
X2 = 0.709459
X3 = -0.258108
```

ELIMINASI GAUSS JORDAN

```
==========SPL==========
1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Metode matriks balikan
4. Metode Cramer
Masukkan jenis penyelesaian SPL: 2
Masukkan matrix melalui terminal (1) atau file (2)
?: 2
Masukkan path file (relatif terhadap folder test):
 3_a.txt
Nilai setiap variabel adalah:
X1 = -0.224324
X2 = 0.182432
X3 = 0.709459
X4 = -0.258108
```

MATRIKS BALIKAN

```
==========SPL==========
1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Metode matriks balikan
4. Metode Cramer
Masukkan jenis penyelesaian SPL: 3
Masukkan matrix melalui terminal (1) atau file (2)
?: 2
Masukkan path file (relatif terhadap folder test):
 3_a.txt
Solusi persamaan:
X1 = -0.224324
X2 = 0.182432
X3 = 0.709459
X4 = -0.258108
```

KAIDAH CRAMER

	==========SPL========== 1. Metode eliminasi Gauss 2. Metode eliminasi Gauss-Jordan 3. Metode matriks balikan 4. Metode Cramer Masukkan jenis penyelesaian SPL: 4 Masukkan matrix melalui terminal (1) atau file (2) ?: 2 Masukkan path file (relatif terhadap folder test): 3_a.txt Solusi persamaan adalah: x1 = -0.2243 x2 = 0.1824 x3 = 0.7095 x4 = -0.2581

4.2

Input	Output
	METODE GAUSS
	==========SPL========== 1. Metode eliminasi Gauss 2. Metode eliminasi Gauss-Jordan 3. Metode matriks balikan 4. Metode Cramer Masukkan jenis penyelesaian SPL: 1 Masukkan matrix melalui terminal (1) atau file (2)?: 2 Masukkan path file (relatif terhadap folder test): 3_b.txt Tidak ada solusi untuk persamaan ini.

4.3 Studi Kasus Interpolasi (1)

Input	Output

0.1 0.003 0.3 0.067 0.5 0.148 0.7 0.248 0.9 0.370 1.1 0.518 1.3 0.699	==========INTERPOLASI LINEAR============ Masukkan data melalui terminal (1) atau melalui file (2)? 1 Masukkan banyak data yang diketahui: 7 Masukkan pasangan nilai x dan y (Minimal 7 pasangan): 0.1 0.003 0.3 0.067 0.5 0.148 0.7 0.248 0.9 0.370 1.1 0.518 1.3 0.697 f(x) = (-0.000000)x^6 + (0.000000)x^5 + (0.026042)x^4 + (0.000000)x^3 + (0.197396)x^2 + (0.240000)x^1 + (-0.022977) Apakah ingin mengaproksimasi nilai? (Y/N): y Masukkan nilai yang ingin diaproksimasi: 0.2 f(0.200000) = 0.032961 Apakah ingin mengaproksimasi nilai? (Y/N): y Masukkan nilai yang ingin diaproksimasi: 0.55 f(0.550000) = 0.171119 Apakah ingin mengaproksimasi nilai? (Y/N): y Masukkan nilai yang ingin diaproksimasi: 0.85 f(0.850000) = 0.337236 Apakah ingin mengaproksimasi nilai? (Y/N): y Masukkan nilai yang ingin diaproksimasi: 1.28 f(1.280000) = 0.677542 Apakah ingin mengaproksimasi nilai? (Y/N):

Studi Kasus Interpolasi (2)

Input	Output
6.567 12624 7 21807 7.258 38391 7.451 54517 7.548 51952 7.839 28228 8.161 35764 8.484 20813 8.709 12408 9 10534 7.516129	Selamat datang di program kalkulator matriks! 1. Penyelesaian SPL 2. Determinan Matriks 3. Invers Matriks 4. Interpolasi Polinomial 5. Regresi Linear Berganda 6. Interpolasi Bicubic Spline 7. Keluar Pilih operasi yang ingin dilakukan: 4 ==========INTERPOLASI LINEAR============ Masukkan data melalui terminal (1) atau melalui file (2)? 2 Masukkan nama file (relatif terhadap test/input): interpol2.txt f(x) = (-225414.392640)x^9 + (16142769.382945)x^8 + (-513520969.632933)x^7 + (9524332040.499626)x^6 + (-113505680765.765870)x^5 + (901392710481.401900)x^4 + (-4770189462014.137000) x^3 + (16221802095816.832000)x^2 + (-32167467366158.840000)x^1 + (28339923520561.160000) Aproksimasi nilai: f(7.516129) = 49052.441406

<table>
<tr>
<td>6.567 12624
7 21807
7.258 38391
7.451 54517
7.548 51952
7.839 28228
8.161 35764
8.484 20813
8.709 12408
9 10534
9.1666667</td>
<td>Selamat datang di program kalkulator matriks!
1. Penyelesaian SPL
2. Determinan Matriks
3. Invers Matriks
4. Interpolasi Polinomial
5. Regresi Linear Berganda
6. Interpolasi Bicubic Spline
7. Keluar
Pilih operasi yang ingin dilakukan: 4

==========INTERPOLASI LINEAR============
Masukkan data melalui terminal (1) atau melalui file (2)? 2
Masukkan nama file (relatif terhadap test/input): interpol2.txt
f(x) = (-225414.392640)x^9 + (16142769.382945)x^8 + (-513520969.632933)x^7 + (9524332040.499626)x^6 + (-113505680765.765870)x^5 + (901392710481.401900)x^4 + (-4770189462014.137000)
x^3 + (16221802095816.832000)x^2 + (-32167467366158.840000)x^1 + (28339923520561.160000)
Aproksimasi nilai: f(9.166667) = -87822.777344</td>
</tr>
<tr>
<td>6.567 12624
7 21807
7.258 38391
7.451 54517
7.548 51952
7.839 28228
8.161 35764
8.484 20813
8.709 12408
9 10534
10.580645</td>
<td>Selamat datang di program kalkulator matriks!
1. Penyelesaian SPL
2. Determinan Matriks
3. Invers Matriks
4. Interpolasi Polinomial
5. Regresi Linear Berganda
6. Interpolasi Bicubic Spline
7. Keluar
Pilih operasi yang ingin dilakukan: 4

==========INTERPOLASI LINEAR============
Masukkan data melalui terminal (1) atau melalui file (2)? 2
Masukkan nama file (relatif terhadap test/input): interpol2.txt
f(x) = (-225414.392640)x^9 + (16142769.382945)x^8 + (-513520969.632933)x^7 + (9524332040.499626)x^6 + (-113505680765.765870)x^5 + (901392710481.401900)x^4 + (-4770189462014.137000)
x^3 + (16221802095816.832000)x^2 + (-32167467366158.840000)x^1 + (28339923520561.160000)
Aproksimasi nilai: f(10.580645) = -1014546867.777344

Apakah Anda ingin menyimpan hasil ke dalam sebuah file (Y/N)?</td>
</tr>
</table>

4.4. Studi Kasus Interpolasi (3)

N = 5

Input	Output

0.4 0.419 0.8 0.507 1.2 0.561 1.6 0.584 2.0 0.577 1.15	==========INTERPOLASI LINEAR============ Masukkan data melalui terminal (1) atau melalui file (2)? 2 Masukkan nama file (relatif terhadap test/input): interpol5.txt f(x) = (-0.003255)x^4 + (0.020833)x^3 + (-0.143229)x^2 + (0.371667)x^1 + (0.292000) Aproksimasi nilai: f(1.150000) = 0.555988

4.5 Studi Kasus Regresi Linear Berganda

Input	Output
	Masukkan nama file (relatif terhadap test/input): RLBtest.txt Hasil Fungsi Dari Regresi Linear Berganda adalah : f(x1,x2,x3) = -3.5078 - 0.0026x1 + 0.0008x2 + 0.1542x3 Hasil taksirannya adalah: f(50.0000,76.0000,29.3000) = 0.9384

4.6 Studi Kasus Bicubic Spline

Input	Output

21 98 125 153 51 101 161 59 0 42 72 210 16 12 81 96 0 0	Selamat datang di program kalkulator matriks! 1. Penyelesaian SPL 2. Determinan Matriks 3. Invers Matriks 4. Interpolasi Polinomial 5. Regresi Linear Berganda 6. Interpolasi Bicubic Spline 7. Keluar Pilih operasi yang ingin dilakukan: 6 Masukkan melalui input manual (1) atau file (2)? : 2 Masukkan nama file: bicubic2.txt Hasil f(0.000000, 0.000000) = 21.000000
21 98 125 153 51 101 161 59 0 42 72 210 16 12 81 96 0.5 0.5	Selamat datang di program kalkulator matriks! 1. Penyelesaian SPL 2. Determinan Matriks 3. Invers Matriks 4. Interpolasi Polinomial 5. Regresi Linear Berganda 6. Interpolasi Bicubic Spline 7. Keluar Pilih operasi yang ingin dilakukan: 6 Masukkan melalui input manual (1) atau file (2)? : 2 Masukkan nama file: bicubic2.txt Hasil f(0.500000, 0.500000) = 87.796875
21 98 125 153 51 101 161 59 0 42 72 210 16 12 81 96 0.25 0.75	Selamat datang di program kalkulator matriks! 1. Penyelesaian SPL 2. Determinan Matriks 3. Invers Matriks 4. Interpolasi Polinomial 5. Regresi Linear Berganda 6. Interpolasi Bicubic Spline 7. Keluar Pilih operasi yang ingin dilakukan: 6 Masukkan melalui input manual (1) atau file (2)? : 2 Masukkan nama file: bicubic2.txt Hasil f(0.250000, 0.750000) = 117.732178
21 98 125 153 51 101 161 59 0 42 72 210 16 12 81 96 0.1 0.9	Selamat datang di program kalkulator matriks! 1. Penyelesaian SPL 2. Determinan Matriks 3. Invers Matriks 4. Interpolasi Polinomial 5. Regresi Linear Berganda 6. Interpolasi Bicubic Spline 7. Keluar Pilih operasi yang ingin dilakukan: 6 Masukkan melalui input manual (1) atau file (2)? : 2 Masukkan nama file: bicubic2.txt Hasil f(0.100000, 0.900000) = 128.575187

4.7 Studi Kasus Bonus Spline Image

Input

```
Input image path:
C:/Users/user/Downloads/borobudur.png
Output image path:
C:/Users/user/Downloads/borobudur_new.png

Scale:
2
Saved!
```

Before	After
	

BAB V
KESIMPULAN

Operasi matriks, seperti perhitungan determinan, penemuan matriks balikan, eliminasi Gauss, dan eliminasi Gauss-Jordan, memiliki aplikasi luas dalam menyelesaikan berbagai masalah. Ambil saja salah satu contohnya adalah pemecahan sistem persamaan linear (SPL). SPL adalah masalah yang umum di dunia nyata dan sering memerlukan solusi yang cepat. Salah satu pendekatan yang efisien adalah dengan menggunakan program komputer yang dapat memanipulasi SPL dalam bentuk matriks.

SPL dapat memiliki tiga jenis solusi: banyak solusi, tidak ada solusi, dan satu solusi unik. Dalam laporan ini, kami membahas empat metode untuk menyelesaikan SPL, yaitu metode Gauss, metode Gauss-Jordan, metode Invers, dan kaidah Cramer. Dalam beberapa kasus, metode Invers dan kaidah Cramer tidak dapat digunakan karena mereka hanya berlaku untuk matriks persegi dengan determinan yang tidak nol. Di sisi lain, metode Gauss dan metode Gauss-Jordan dapat menghasilkan solusi berupa persamaan parametrik.

SPL kemudian dapat digunakan secara nyata dalam aplikasi Interpolasi Polinom, Interpolasi *Bicubic Spline*, dan Regresi Linear Berganda. Interpolasi Polinom digunakan untuk menghasilkan persamaan polinom dari sekumpulan titik data yang dikenal, memungkinkan prediksi nilai di luar titik-titik tersebut. Dalam BAB IV, kami menggunakan interpolasi polinom untuk memprediksi kasus Covid-19 pada tanggal tertentu dan untuk menyederhanakan fungsi matematika. Interpolasi *Bicubic Spline* digunakan untuk memprediksi nilai pada titik-titik tertentu dengan mempertimbangkan 16 nilai yang ada di sekitarnya. Sementara itu, Regresi Linear Berganda digunakan untuk memprediksi nilai yang dipengaruhi oleh beberapa variabel. Dalam BAB IV, kami menggunakan Regresi Linear Berganda untuk memprediksi konsentrasi Nitrous Oxide berdasarkan kelembaban udara, tekanan, dan suhu.

BAB VI
REFLEKSI

Melalui tugas besar ini, kami bisa lebih memahami lebih mengenai materi SPL yang diajarkan di kelas serta mengaplikasikannya dalam wujud yang konkret. Kami juga dikenalkan dengan “tugas yang melimpah dan lautan ujian” yang katanya belum seberapa jika dibandingkan dengan lautan-lautan ke depannya. Kami juga bahkan ditantang untuk mewujudkan semuanya itu dengan bahasa dan paradigma pemrograman yang belum kami kenali. Namun, kami bangga karena telah menghadapi semua itu layaknya seorang kesatria yang memilih untuk mencoba dulu yang terbaik dibandingkan mengeluh dan menyerah. Kami menyadari bahwa kami tentu masih banyak kekurangan dan untuk itu kami ingin meminta maaf atas hal tersebut. Namun, layaknya sebuah proses belajar yang walau banyak jatuh bangunnya harus tetap memberikan esensi, kami merasa telah berhasil menyerap esensi dari Tugas Besar Aljabar Linier dan Geometri dalam wujud peningkatan tingkat pemahaman kami mengenai materi SPL, Determinan, dan Aplikasinya.

Untuk Tuhan, Bangsa, dan Almamater. Hidup Informatika!

DAFTAR PUSTAKA

Stack Overflow. 2012. Java: *Get pixel array from image*. https://stackoverflow.com/questions/6524196/java-get-pixel-array-from-image

Wikipedia. (n.d.). *Bicubic interpolation*. https://en.wikipedia.org/wiki/Bicubic_interpolation

Abdullah, Dahlan. 2018. Application of Interpolation Image by using Bi-Cubic Algorithm. *IOP Conference Series: Materials Science and Engineering*, *1114*(1), 012066. https://iopscience.iop.org/article/10.1088/1742-6596/1114/1/012066/pdf

Rowe, D. P. 2018. *BiLinear, Bicubic, and In Between Spline Interpolation*. Marquette University. https://www.mssc.mu.edu/~daniel/pubs/RoweTalkMSCS_BiCubic.pdf

CodeJava. 2019. *How to resize images in Java*. https://www.codejava.net/java-se/graphics/how-to-resize-images-in-java

Rinaldi Munir. 2020. *Sistem Persamaan Linier (SPL): Metode Eliminasi Gauss-Jordan*. Institut Teknologi Bandung. https://informatika.stei.itb.ac.id/~rinaldi.munir/AljabarGeometri/2020-2021/Algeo-05-Sistem-Persamaan-Linier-2.pdf

I Made Yuliara. 2016. *Modul Regresi Linier Berganda*. Universitas Udayana. https://simdos.unud.ac.id/uploads/file_pendidikan_1_dir/5f0221d2b0bb7ced1d61798fab7f4ad3.pdf

Anonymous. 2020. *Polynomial Interpolation*. Indian Statistical Institute. https://www.isical.ac.in/~arnabc/numana/interpol1.html