

# **Laporan Tugas Besar I**

## **IF2123 Aljabar Linier dan Geometri**



**Kelompok 20 - Panci Bolong:**

Raden Francisco Trianto B.	13522091
Matthew Vladimir Hutabarat	13522093
Suthasoma Mahardhika Munthe	13522098

**INSTITUT TEKNOLOGI BANDUNG**  
**2023**

# DAFTAR ISI

<b>Daftar Isi .....</b>	<b>ii</b>
<b>Bab I Deskripsi Masalah .....</b>	<b>1</b>
<b>Bab II Teori Singkat.....</b>	<b>7</b>
2.1 Metode Operasi Baris Elementer .....	7
2.2 Metode Eliminasi Gauss .....	7
2.3 Metode Eliminasi Gauss-Jordan .....	7
2.4 Determinan.....	7
2.5 Matrix Balikan .....	8
2.6 Matrix Kofaktor .....	8
2.7 Adjoin Matrix.....	9
2.8 Kaidah Cramer .....	9
2.9 Interpolasi Polinom.....	9
2.10 Interpolasi Bicubic .....	10
2.11. Regresi Linear Berganda.....	10
<b>Bab III Implementasi.....</b>	<b>12</b>
<b>Bab IV Eksperimen.....</b>	<b>24</b>
4.1 Sistem Persamaan Linier.....	24
a. Matriks Bersolusi Tunggal .....	24
b. Matriks Tidak Bersolusi .....	26
c. Matriks Bersolusi Parameter .....	27
d. Invalid .....	28
4.2 Determinan.....	28
a. Metode Ekspansi Kofaktor.....	28
b .Metode Reduksi Baris dengan OBE .....	29
4.3 Matriks Balikan.....	30
a. Matriks Balikan .....	30
b .Metode Adjoin .....	30
4.4 Interpolasi Polinom .....	31
4.5 Regresi Linier Berganda .....	37
4.6 Bicubic Spline Interpolation .....	37
4.7 Aplikasi <i>Bicubic Spline Interpolation</i> - Peningkatan Kualitas Gambar (ImageBSI) .....	39
<b>Bab V Kesimpulan .....</b>	<b>41</b>
<b>Referensi .....</b>	<b>42</b>

# BAB I

## DESKRIPSI MASALAH

### 1. ABSTRAKSI

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

$$\begin{bmatrix} 0 & \mathbf{2} & 1 & -1 \\ 0 & 0 & \mathbf{3} & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} 0 & \mathbf{1} & 0 & -\frac{2}{3} \\ 0 & 0 & \mathbf{1} & \frac{1}{3} \\ 0 & 0 & 0 & 0 \end{bmatrix}.$$

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

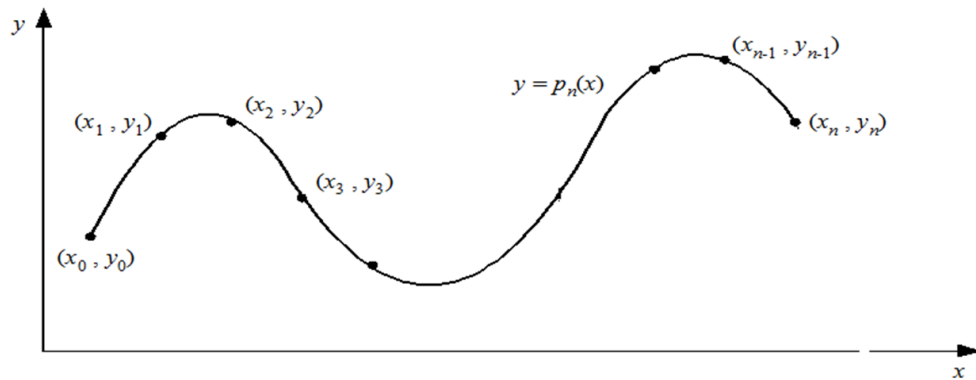
Di dalam Tugas Besar 1 ini, Anda diminta membuat satu atau lebih *library* aljabar linier dalam Bahasa Java. Library tersebut berisi fungsi-fungsi seperti eliminasi Gauss, eliminasi Gauss-Jordan, menentukan balikan matriks, menghitung determinan, kaidah *Cramer* (kaidah *Cramer* khusus untuk SPL dengan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

Beberapa tulisan cara membuat library di Java:

1. <https://www.programcreek.com/2011/07/build-a-java-library-for-yourself/>
2. <https://developer.ibm.com/tutorials/j-javalibrary/>
3. <https://stackoverflow.com/questions/3612567/how-to-create-my-own-java-library>

### 2. Interpolasi Polinomial

Persoalan interpolasi polinom adalah sebagai berikut: Diberikan  $n+1$  buah titik berbeda,  $(x_0, y_0), (x_1, y_1), \dots, (x_n, y_n)$ . Tentukan polinom  $p_n(x)$  yang menginterpolasi (melewati) semua titik-titik tersebut sedemikian rupa sehingga  $y_i = p_n(x_i)$  untuk  $i = 0, 1, 2, \dots, n$ .



**Gambar 2.** Ilustrasi beberapa titik yang diinterpolasi secara polinomial.

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

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

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

Solusi sistem persamaan linier ini, yaitu nilai  $a_0, a_1, \dots, a_n$ , diperoleh dengan menggunakan metode eliminasi Gauss yang sudah anda pelajari. Sebagai contoh, misalkan diberikan tiga buah titik yaitu  $(8.0, 2.0794)$ ,  $(9.0, 2.1972)$ , dan  $(9.5, 2.2513)$ . Tentukan polinom interpolasi kuadratik lalu estimasi nilai fungsi pada  $x = 9.2$ . Polinom kuadratik berbentuk  $p_2(x) = a_0 + a_1x + a_2x^2$ . Dengan menyulihkan ketiga buah titik data ke dalam polinom tersebut, diperoleh sistem persamaan linier yang terbentuk adalah

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

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

### 3. Regresi Linier Berganda

Regresi Linear (akan dipelajari lebih lanjut di Probabilitas dan Statistika) merupakan salah satu metode untuk memprediksi nilai selain menggunakan Interpolasi Polinom. Meskipun sudah ada persamaan jadi untuk menghitung regresi linear sederhana, terdapat persamaan umum dari regresi linear yang bisa digunakan untuk regresi linear berganda, yaitu.

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

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

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

Sistem persamaan linier tersebut diselesaikan dengan menggunakan metode eliminasi Gauss.

### 4. Bicubic Spline Interpolation

*Bicubic spline interpolation* adalah metode interpolasi yang digunakan untuk mengaproksimasi fungsi di antara titik-titik data yang diketahui. *Bicubic spline interpolation* melibatkan konsep *spline* dan konstruksi serangkaian polinomial kubik di dalam setiap sel segi empat dari data yang diberikan. Pendekatan ini menciptakan permukaan yang halus dan kontinu, memungkinkan untuk perluasan data secara visual yang lebih akurat daripada metode interpolasi linear.

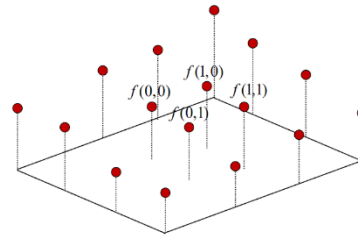
Dalam pemrosesan menggunakan interpolasi *bicubic spline* digunakan 16 buah titik, 4 titik referensi utama di bagian pusat, dan 12 titik di sekitarnya sebagai aproksimasi turunan dari keempat titik referensi untuk membangun permukaan bikubik. Bentuk pemodelannya adalah sebagai berikut.

Normalization:  $f(0,0), f(1,0)$

$f(0,1), f(1,1)$

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

Solve:  $a_{ij}$



**Gambar 3.** Pemodelan interpolasi *bicubic spline*.

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

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

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

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

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

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

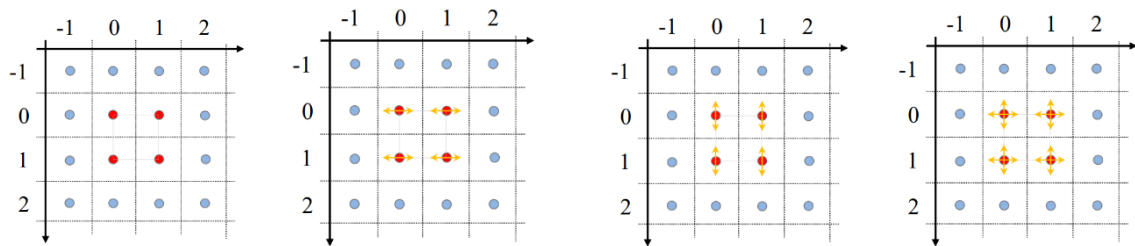
$$y = Xa$$

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

Perlu diketahui bahwa elemen pada matriks  $X$  adalah nilai dari setiap komponen koefisien  $a_{ij}$  yang diperoleh dari persamaan fungsi maupun persamaan turunan yang telah dijelaskan sebelumnya. Sebagai contoh, elemen matriks  $X$  pada baris 8 kolom ke 2 adalah

koefisien dari  $a_{10}$  pada ekspansi sigma untuk  $f(1, 1)$  sehingga diperoleh nilai konstanta  $1 \times 1^{1-1} \times 1^0 = 1$ , sesuai dengan isi matriks  $X$ .

Nilai dari vektor  $a$  dapat dicari dari persamaan  $y = Xa$ , lalu vektor  $a$  tersebut digunakan sebagai nilai variabel dalam  $f(x, y)$ , sehingga terbentuk fungsi interpolasi bicubic sesuai model. Tugas Anda pada studi kasus ini adalah membangun persamaan  $f(x, y)$  yang akan digunakan untuk melakukan interpolasi berdasarkan nilai  $f(a, b)$  dari masukan matriks  $4 \times 4$ . Nilai masukan  $a$  dan  $b$  berada dalam rentang  $[0, 1]$ . Nilai yang akan diinterpolasi dan turunan berarah disekitarnya dapat diilustrasikan pada titik berwarna merah pada gambar di bawah.



**Gambar 4.** Nilai fungsi yang akan diinterpolasi pada titik merah, turunan berarah terhadap sumbu  $x$ , terhadap sumbu  $y$ , dan keduanya (kiri ke kanan).

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

## 5. Peningkatan Kualitas Gambar dengan Interpolasi bicubic Spline

Seperti yang telah dijelaskan sebelumnya bahwa interpolasi *bicubic spline* dapat digunakan untuk menciptakan permukaan yang halus pada gambar. Oleh karena itu, selain persamaan dasar  $y = Xa$  yang telah dijabarkan, persamaan ini juga dapat menggunakan data sebuah citra untuk menciptakan kualitas gambar yang lebih baik. Misalkan  $I(x, y)$  merupakan nilai dari suatu citra gambar pada posisi  $(x, y)$ , maka dapat digunakan persamaan nilai dan persamaan turunan berarah sebagai berikut.

$$f(x, y) = I(x, y)$$

$$f_x(x, y) = [I(x+1, y) - I(x-1, y)] / 2$$

$$f_y(x, y) = [I(x, y+1) - I(x, y-1)] / 2$$

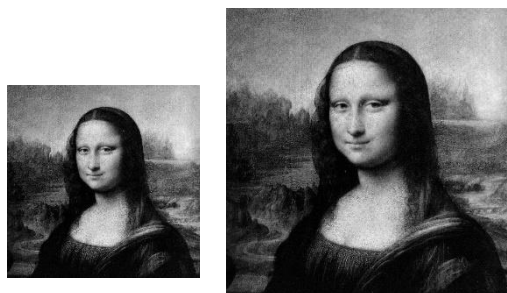
$$f_{xy}(x, y) = [I(x+1, y+1) - I(x-1, y) - I(x, y+1) - I(x, y-1)] / 4$$

Sistem persamaan tersebut dapat dipetakan menjadi sebuah matriks (dalam hal ini matriks  $D$ ) dengan gambaran lengkap seperti yang tertera di bawah.

$$y = DI$$

$$\begin{bmatrix} f(0,0) \\ f(1,0) \\ f(0,1) \\ f(1,1) \\ f_x(0,0) \\ f_x(1,0) \\ f_x(0,1) \\ f_x(1,1) \\ f_y(0,0) \\ f_y(1,0) \\ f_y(0,1) \\ f_y(1,1) \\ f_{xy}(0,0) \\ f_{xy}(1,0) \\ f_{xy}(0,1) \\ f_{xy}(1,1) \end{bmatrix} = \frac{1}{4} \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 4 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -2 & 0 & 2 & 0 & 0 & 0 & 4 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -2 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -2 & 0 & 2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -2 & 0 & 2 & 0 & 0 & 0 & 0 \\ 0 & -2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -2 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -2 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 \\ 0 & -1 & 0 & 0 & -1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & -1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & -1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & -1 & 1 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} I(-1,-1) \\ I(0,-1) \\ I(1,-1) \\ I(2,-1) \\ I(-1,0) \\ I(0,0) \\ I(1,0) \\ I(2,0) \\ I(-1,1) \\ I(0,1) \\ I(1,1) \\ I(2,1) \\ I(-1,2) \\ I(0,2) \\ I(1,2) \\ I(2,2) \end{bmatrix}$$

Dengan menggunakan kedua persamaan nilai  $y$  yang telah disebutkan dan dibahas sebelumnya, dapatkan nilai  $a$  yang lebih baik dan akurat dalam pemrosesan citra gambar, kemudian gunakan nilai dan persamaan  $f(x, y)$  yang terbentuk untuk memperbaiki kualitas citra gambar monokrom pasca perbesaran dengan skala tertentu dengan melakukan interpolasi *bicubic spline*. Berikut adalah contohnya.



**Gambar 5.** Sebuah citra gambar asal (kiri) dan hasil perbesarannya dengan skala 1.5 (kanan).

Untuk bonus ini, buatlah matriks  $D$  menggunakan persamaan citra gambar yang ada (tidak *hardcode*) serta gunakan kembali persamaan  $y$  yang sebelumnya ( $y = Xa$ ) dan korelasikan dengan persamaan  $y = DI$  untuk mendapatkan nilai  $a$  yang lebih tepat untuk membangun persamaan  $f(x, y)$ . Tambahkan pula masukan berupa skala perbesaran gambar sesuai keinginan pengguna.



## BAB II

### TEORI SINGKAT

#### 2.1 Metode Operasi Baris Elementer

Metode Operasi Baris Elementer merupakan cara yang digunakan untuk mendapatkan solusi dari beberapa persamaan linear atau invers yang nantinya banyak dipakai untuk mencari Determinan, Interpolasi, Regresi, dll. Operasi ini dilakukan terus menerus pada suatu Matrix Augmented hingga terbentuk matrix eselon baris atau matrix eselon baris tereduksi sesuai dengan solusi yang dihasilkan. Terdapat 3 tipe solusi yang dihasilkan jika matrix sudah berbentuk eselon yaitu, solusi unik (1 solusi), solusi banyak (lebih dari 1 solusi), dan tidak punya solusi. Beberapa OBE terhadap matrix yaitu:

1. Mengalikan sebuah baris dengan konstanta tidak nol.
2. Dua baris bertukar posisi.
3. Operasi tambah dan kurang suatu baris ke baris lainnya.

#### 2.2 Metode Eliminasi Gauss

Metode Eliminasi Gauss mengubah beberapa persamaan linear menjadi bentuk matrix kemudian diubah ke dalam bentuk eselon baris menggunakan OBE. Hal ini dilakukan agar diakhir lebih mudah menggunakan metode substitusi dan mendapatkan solusi persamaan.

$$\begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} & b_1 \\ a_{21} & a_{22} & \cdots & a_{2n} & b_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} & b_n \end{pmatrix} \sim_{\text{OBE}} \begin{pmatrix} 1 & * & * & \cdots & * & * \\ 0 & 1 & * & \cdots & * & * \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \vdots & 1 & * \end{pmatrix}$$

#### 2.3 Metode Eliminasi Gauss-Jordan

Metode ini sama seperti Eliminasi Gauss hanya saja perbedaannya ada di bentuk matrix akhirnya menjadi eselon baris tereduksi yang mana dibelakang angka 1 tidak ada lagi konstanta selain kolom terakhir yang lebih besar dari 0. Hasil akhir metode ini lebih memudahkan kita untuk mendapatkan solusi tanpa menggunakan metode substitusi.

$$\begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} & b_1 \\ a_{21} & a_{22} & \cdots & a_{2n} & b_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} & b_n \end{pmatrix} \sim_{\text{OBE}} \begin{pmatrix} 1 & 0 & 0 & \cdots & 0 & * \\ 0 & 1 & 0 & \cdots & 0 & * \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \vdots & 0 & * \end{pmatrix}$$

#### 2.4 Determinan

Determinan merupakan bagian dari matrix dan dapat dihitung nilainya dari unsur unsur matrix. Determinan hanya bisa dihitung jika matrixnya persegi. Cara mendapatkan nilainya dapat dilakukan dengan berbagai metode yaitu metode reduksi baris, metode ekspansi kofaktor, dan metode sarrus. Determinan dapat dipakai untuk menyelesaikan SPL dan mendapatkan solusi.

Pada metode reduksi baris, digunakan OBE secara terus menerus hingga mendapatkan matrix segitiga, (segitiga atas atau bawah).

$$\begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{pmatrix} \sim_{\text{OBE}} \begin{pmatrix} a'_{11} & a'_{12} & \cdots & a'_{1n} \\ 0 & a'_{22} & \cdots & a'_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & a'_{nn} \end{pmatrix}$$

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

$$\begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{pmatrix} \sim \mathbf{OBE} \sim \begin{pmatrix} a'_{11} & 0 & \cdots & a'_{1n} \\ a'_{21} & 0 & \cdots & a'_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ a'_{n1} & a'_{n2} & \cdots & a'_{nn} \end{pmatrix}$$

*Matrix segitiga Atas (B)*

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

Pada metode ekspansi kofaktor, misal  $A_{ij}$  merupakan matrix bagian dari matrix A yang diperoleh dengan menghilangkan baris ke - i dan kolom ke - j lalu  $C_{ij} = (-1)^{i+j} M_{ij}$  dengan  $M_{ij}$  merupakan  $A_{ij}$ . Untuk rumus determinannya sebagai berikut

$$\det(A) = \sum_{i=1}^n a_{ij} \cdot C_{ij}, \text{ untuk sembarang baris } i=1,2,3,\dots,n$$

$$\det(A) = \sum_{j=1}^n a_{ij} \cdot C_{ij}, \text{ untuk sembarang kolom } j=1,2,3,\dots,n$$

## 2.5 Matrix Balikan

Matrix Balikan adalah invers dari suatu Matrix yang berdimensi persegi. Matrix ini bisa didapatkan dengan menggunakan rumus  $M^{-1} = \frac{1}{\det M} \text{Adj}(A)$  atau memakai metode Gauss Jordan yang mengubahnya menjadi matrix eselon tereduksi. Ingat kalau  $A(A^{-1}) = (A^{-1})A = I$

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

Sebagai perbandingan menggunakan rumus dengan menggunakan Gauss Jordan.

1. Menggunakan rumus  $M^{-1} = \frac{1}{\det M} \text{Adj}(A)$

$$A = \begin{pmatrix} 3 & 1 \\ 5 & 2 \end{pmatrix}$$

$$\det(A) = 3 \cdot 2 - 1 \cdot 5 = 1$$

$$A^{-1} = \frac{1}{1} \begin{pmatrix} 2 & -1 \\ -5 & 3 \end{pmatrix} = \begin{pmatrix} 2 & -1 \\ -5 & 3 \end{pmatrix}$$

2. Menggunakan rumus Gauss Jordan  $(A|I) \sim \mathbf{GJ} \sim (I|A^{-1})$

$$\left( \begin{array}{cc|cc} 3 & 1 & 1 & 0 \\ 5 & 2 & 0 & 1 \end{array} \right) \sim \mathbf{GJ} \sim \left( \begin{array}{cc|cc} 1 & 0 & 2 & -1 \\ 0 & 1 & -5 & 3 \end{array} \right)$$

$$A^{-1} = \begin{pmatrix} 2 & -1 \\ -5 & 3 \end{pmatrix}$$

## 2.6 Matrix Kofaktor

Kofaktor merupakan hasil perkalian minor dengan suatu angka yang besarnya mengikuti suatu urutan yaitu  $(-1)^{i+j}$  dimana i adalah baris dan j adalah kolom. Matrix kofaktor merupakan matrix yang terdiri dari kofaktor-kofaktor matrix itu sendiri. Sebagai contoh, Matrix  $A = \begin{bmatrix} -1 & 3 \\ 4 & -5 \end{bmatrix}$ , maka kofaktor-kofaktornya adalah  $C_{11} = -5$ ,  $C_{12} = -4$ ,  $C_{21} = -3$ , dan  $C_{22} = -1$ . Maka Matrix kofaktor

$A = \begin{bmatrix} -5 & -4 \\ -3 & -1 \end{bmatrix}$ . Matrix kofaktor ini dibutuhkan jika mau mencari determinan dengan cara ekspansi kofaktor.

## 2.7 Adjoin Matrix

Adjoin matrix merupakan transpose dari matriks kofaktor. Transpose sendiri merupakan pertukaran elemen pada baris menjadi kolom dan kolom menjadi baris. Adjoin matrix sering dipakai untuk menentukan nilai invers matrix.

$$\begin{aligned} \text{Matrix } A &= \begin{bmatrix} -1 & 3 \\ 4 & -5 \end{bmatrix} \\ \text{Matrix Kofaktor } A &= \begin{bmatrix} -5 & -4 \\ -3 & -1 \end{bmatrix} \\ \text{Adj}(A) &= \begin{bmatrix} -5 & -3 \\ -4 & -1 \end{bmatrix} \end{aligned}$$

## 2.8 Kaidah Cramer

Kaidah Cramer digunakan untuk menyelesaikan persamaan linear dengan n persamaan dalam n variable. Metode ini memanfaatkan determinan dalam mencari solusi persamaan. Misalkan terdapat 2 sistem persamaan linear 3 variabel.  $a_1x + b_1y = c_1$  dan  $a_2x + b_2y = c_2$ . Kita bisa menggunakan metode eliminasi untuk mendapatkan nilai x sebagai berikut

$$\begin{array}{rcl} sa_1x + b_1y = c_1 & \xrightarrow{xb_2} & a_1b_2x + b_1b_2y = c_1b_2 \\ a_2x + b_2y = c_2 & \xrightarrow{xb_1} & a_2b_1x + b_1b_2y = c_2b_1 \quad - \\ \hline (a_1b_2 - a_2b_1)x & = & c_1b_2 - c_2b_1 \\ x & = & \frac{c_1b_2 - c_2b_1}{a_1b_2 - a_2b_1} \end{array}$$

Hasil dari x ini dapat dihubungkan dengan determinan, anggap  $x = \begin{pmatrix} a_1 & b_1 \\ a_2 & b_2 \end{pmatrix}$  sehingga

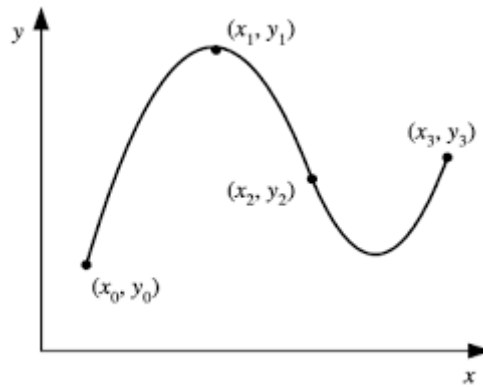
$$x = \frac{D_x}{D} = \frac{\begin{vmatrix} c_1 & b_1 \\ c_2 & b_2 \end{vmatrix}}{\begin{vmatrix} a_1 & b_1 \\ a_2 & b_2 \end{vmatrix}}$$

Dengan cara yang sama kita bisa juga mendapatkan  $y = \frac{D_y}{D}$  dengan menggunakan metode eliminasi x sehingga dapat disimpulkan solusi persamaan linear adalah

$$x_1 = \frac{D_{x_1}}{D}, x_2 = \frac{D_{x_2}}{D}, \dots, x_n = \frac{D_{x_n}}{D}$$

## 2.9 Interpolasi Polinom

Metode Interpolasi Polinom adalah metode yang dipakai untuk menaksir nilai yang tidak diketahui dengan cara mengambil beberapa data nilai yang diketahui dan mengolahnya menjadi suatu polinom yang akan melewati semua titik yang diketahui. Untuk menggunakan metode interpolasi polinom dibutuhkan data seminimalnya  $n + 1$  titik untuk dapat membuat polinom interpolasi  $p_n(x)$ , setelah polinom ditemukan dapat ditaksir nilai y di sembarang titik didalam selang  $[x_0, x_n]$



## 2.10 Interpolasi Bicubic

Metode Interpolasi Bicubic merupakan metode yang digunakan untuk mengaproksimasi nilai fungsi diantara titik titik data yang diketahui. Metode ini melibatkan konsep spline dan konstruksi serangkaian polinomial kubik. Dengan metode ini kita bisa melakukan pendekatan untuk menciptakan permukaan yang lebih halus dan kontinu. Salah satu penerapan interpolasi bicubic untuk scaling image, dengan menggunakan metode ini kita bisa memperbesar gambar dan membuat gambar lebih jernih dari sebelumnya.

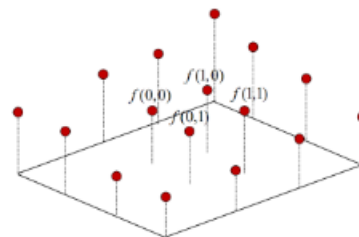
Untuk menggunakan metode ini dibutuhkan 16 buah titik yang terdiri 4 titik referensi utama pada bagian pusat dan sisanya 12 titik di sekitarnya sebagai aproksimasi turunan dari keempat titik referensi utama. Untuk mempermudah pemodelannya bisa dilihat sebagai berikut.

Normalization:  $f(0,0), f(1,0)$

$f(0,1), f(1,1)$

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

Solve:  $a_{ij}$



## 2.11. Regresi Linear Berganda

Metode Regresi Linear berganda merupakan metode yang dapat menafsir nilai berdasarkan data data yang diproses terlebih dahulu. Tidak seperti Metode Interpolasi Polinom yang wajib menginterpolasi semua titik titik data yang diberikan dan membuat rumus polinom yang bisa berderajat lebih dari satu . Di Metode regresi ini kita hanya mengambil “jalur tengah” dari data - data yang diberikan.

Dalam metode regresi linear berganda terdapat variable bebas dan variable terikat. Variabel bebas menjelaskan hubungan/relasi/pengaruhnya terhadap variable terikat , seperti namanya nilai dari variable ini bebas sedangkan variable terikat merupakan variable yang terikat oleh variable bebas. Adapun rumus umum untuk regresi linear berganda yakni

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

$$x_{1i}, x_{2i}, \cdots, x_{ki} = \text{variable terikat}$$

$$y_i = \text{variable bebas}$$

Untuk mendapatkan variable bebas , terikat beserta konstanta konstantanya, kita bisa menggunakan *Normal Estimation Equation for Multiple Linear Regression* sebagai berikut:

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

## BAB III

### IMPLEMENTASI

#### 3.1 Folder Operations

##### a. Matrix

Class Matrix berisi atribut, konstruktor, fungsi, dan prosedur yang digunakan untuk membuat dan mengolah matrix yang nantinya banyak dipakai untuk memecahkan berbagai permasalahan

- **Atribut**

Atribut	Deskripsi
Matrix	Variable bertipe matrix
rowEff	Variable yang bertipe integer yang menampung banyak baris
colEff	Variable yang bertipe integer yang menampung banyak kolom

- **Konstruktor**

Atribut	Deskripsi
Matrix (int rowEff, int colEff)	Konstruktor untuk membuat Matrix yang memiliki kolom sebanyak variable colEff dan baris sebanyak variable rowEff
Matrix ()	Konstruktor untuk membuat Matrix sebagai deklarasi dengan baris dan kolom efektif tidak ada / 0
Matrix ( Matrix matrix )	Konstruktor untuk membuat Matrix yang hasilnya sama persis dengan matrix inputan
getRowEff ()	Konstruktor ini untuk mengakses banyak baris efektif
setRowEff ( int newRow )	Konstruktor ini untuk mengubah nilai banyak baris efektif
getColEff ()	Konstruktor ini untuk mengakses banyak kolom efektif
setColEff ( int newCol )	Konstruktor ini untuk mengubah nilai banyak kolom efektif
getElmt ( int i, int j )	Konstruktor ini untuk mengakses element untuk baris i dan kolom j
setElmt( double val, int i, int j )	Konstruktor ini untuk mengubah element yang berada di baris i dan kolom j
getRow ( int i )	Konstruktor ini untuk mengakses semua element yang ada di baris i
setRow ( double[ ] newRow, int i )	Konstruktor ini untuk mengubah nilai semua element yang ada di baris i dengan baris baru
getCol ( int j )	Konstruktor ini untuk mengakses semua element yang ada di kolom j

setCol ( double[ ] newCol, int j )	Konstruktor ini untuk mengubah nilai semua element yang ada di kolom j dengan kolom baru
------------------------------------	--

### • Fungsi dan Prosedure

Fungsi / Prosedur	Deskripsi
toString ( )	Fungsi yang menampilkan baris dan kolom efektif
readMatrix ( int row, int col )	Procedure yang membaca matrix dari user dengan banyak baris sesuai dengan nilai variable baris dan banyak kolom dengan nilai variable kolom
displayMatrix ( )	Procedure yang menampilkan matrix
isSquare ( )	Fungsi yang memberikan true jika dimensi matrix berukuran sama untuk baris dan kolomnya
isIdxValid ( int i, int j )	Fungsi yang memberikan true jika variable i dan j merupakan bagian dari index baris dan kolom yang adad di matrix
isIdxEff ( int i, int j )	Fungsi yang memberikan true jika variable i dan j merupakan bagian dari index baris dan kolom efektif di matrix
countElmt ( )	Fungsi yang mengembalikan banyak element yang efektif di matrix
createString ( int i, int j, int length )	Fungsi yang mengubah tipe data untuk elemen dengan index i dan j dari double menjadi string yang lebih pendek angka dibelakang koma
minorMatrix ( int row, int col )	Fungsi yang mengembalikan Matrix minor yang nantinya digunakan untuk mencari kofaktor
Determinant ( )	Fungsi yang mengembalikan nilai determinant dari Matrix
copyMatrix ( Matrix input )	Procedure yang mengembalikan matrix baru yang isi matrix baru identik dengan matrix input user
transposeMatrix ( )	Procedure yang mengubah baris matrix menjadi kolom dan kolom menjadi baris
rTransposeMatrix ( )	Fungsi yang mengubah baris matrix menjadi kolom dan kolom menjadi baris
multiplyMatrixByConst ( double k )	Procedure yang mengolah nilai semua element matrix dengan mengalikan variable k
kofaktorMatrix ( )	Fungsi yang mengembalikan kofaktor dari matrix dengan pre kondisi, matrix harus berukuran persegi
inversMatrix ( )	Fungsi yang mengembalikan matrix yang merupakan invers dari matrix yang ada
multiplyMatrix ( )	Fungsi yang mengalikan matrix dengan matrix baru dan menampung datanya di matrix baru

Atribut	Deskripsi
OBE ( )	Konstruktor untuk deklarasi awal tipe data OBE
OBE ( int row, int col )	Konstruktor untuk deklarasi awal tipe data OBE dengan jumlah baris matrix sebanyak variable row dan kolom sebanyak variable kolom
OBE ( OBE newOBE )	Konstruktor untuk deklarasi awal tipe data OBE dengan mengembalikan OBE baru yang sama persis dengan OBE inputan
getCopyAugmented ( )	Konstruktor yang mengembalikan matrix baru sebagai deklarasi matrix
setAugmented ( Matrix m )	Konstruktor yang mengubah semua matrix augmented ke matrix baru
getAllCol ( int j )	Konstruktor yang dapat mengakses semua kolom di variable j
getMatrixRow ( )	Konstruktor yang dapat mengakses jumlah baris efektif
setMatrixRow ( int i )	Konstruktor yang mengubah besar nilai jumlah baris efektif menjadi senilai variable i
getMatrixCol ( )	Konstruktor yang dapat mengakses jumlah kolom efektif
setMatrixCol ( int j )	Konstruktor yang mengubah besar nilai jumlah kolom efektif menjadi variable j
getMElmt ( int i, int j )	Konstruktor yang dapat mengakses elemen yang berada di baris i dan kolom j
setMElmt ( double val, int i, int j )	Konstruktor yang mengubah elemen yang berada di baris i dan kolom j menjadi variable val
getIndexMain ( int i )	Konstruktor yang mengembalikan nilai dari elemen index main pada index i
setIndexMain ( int val, int i )	Konstruktor yang mengubah nilai dari elemen index main pada index i menjadi variable val
getSolutions ( int i )	Konstruktor yang mengembalikan solusi dari list solusi pada index i
setSolutions ( double val, int i )	Konstruktor yang mengubah nilai solusi dari list solusi pada index i dengan variable val
getParameter ( int i )	Konstruktor yang mengakses list parameter pada index ke – i
setParameter ( String s, int i )	Konstruktor yang mengubah nilai parameter pada index I dengan variable s
setSolusiUnik ( boolean cond )	Konstruktor yang megubah variable solusiUnik menjadi variable cond
setNoSolusi ( boolean cond )	Konstruktor yang mengubah variable NoSolusi menjadi variable cond
getStep ( )	Konstruktor yang mengembalikan variable stepByStep



getNoSolusi ( )	Konstruktor yang mengembalikan variable noSolusi
getSolusiUnik ( )	Konstruktor yang mengembalikan variable solusiUnik
setSolusi ( )	Konstruktor yang mengubah nilai list pada variable solutions

#### **b. OBE**

Class OBE berisi atribut, konstruktor, fungsi , dan prosedur yang digunakan untuk membuat dan mengolah matrix untuk melakukan operasi OBE yang nantinya banyak dipakai untuk memecahkan berbagai permasalahan

- **Atribut**

Atribut	Deskripsi
Augmented	Variable dengan tipe data Matrix
indexMain	Variable dengan tipe data list integer untuk menampung index utama pada matrix
Solusi	Variable dengan tipe data list double untuk menampung solusi
Parameter	Variable dengan tipe data list string untuk menampung parameter yang ada
solusiUnik	Variable dengan tipe data boolean untuk mengetahui kondisi solusi apakah unik atau tidak
noSolusi	Variable dengan tipe data boolean untuk mengetahui kondisi solusi apakah ada atau tidak
stepByStep	Variable dengan tipe data sstring untuk menampung step step penyelesaian pada OBE

- **Konstruktor dan Selektor**

- **Fungsi dan Prosedure**

Fungsi / Prosedur	Deskripsi
toString ( )	Fungsi untuk menampilkan status solusiUnik dan tidak ada solusi
formatDouble ( int i, int j, int length )	Fungsi untuk mengubah format double menjadi lebih pendek dan mengembalikannya dengan tipe data string
printAugmented ( )	Procedure yang menampilkan matrix Augmented
printIMain ( )	Procedure yang menampilkan isi lisit IndexMain
isSquare ( )	Fungsi yang mengembalikan true jika dimensi matrix berbentuk persegi

isIdxValid( int i, int j )	Fungsi yang mengembalikan true jika index i dan j ada pada index index di matrix
isIdxEff ( int i, int j )	Fungsi yang mengembalikan true jika index i dan j ada pada index index yang efektif di matrix
determinant ( )	Fungsi yang mengembalikan nilai determinan matrix
transposeMatrix ( )	Procedure yang mengubah kolom matrix menjadi baris dan baris menjadi kolom
rTransposeMatrix ( )	Fungsi yang mengubah kolom matrix menjadi baris dan baris menjadi kolom
multiplyMatrixByConst ( double k )	Procedure yang memproses semua nilai elemen dengan mengalikan dengan variable k
kofakMatrix ( )	Fungsi yang mengembalikan kofaktor matrix
inversMatrix ( )	Fungsi yang mengembalikan matrix menjadi inversnya
multiplyMatrix ( Matrix m )	Fungsi yang memproses perkalian 2 matrix
isContinue ( )	Fungsi yang mengecek apakah operasi OBE perlu dilanjutkan atau tidak
addAugmentedToStep ( int length )	Procedure yang akan menambahkan matriks Augmented ke dalam atribut stepByStep
addSubtractToStep ( int row1, int row2, double left, double right )	Procedure yang akan menambahkan proses pengurangan setiap baris saat melakukan OBE ke dalam atribut stepByStep
addStringToStep ( String add )	Procedure yang akan menambahkan String sesuai keinginan ke dalam atribut stepByStep
addNewLineToStep ( )	Procedure yang menambahkan new line pada variable stepByStep
addSwapToStep ( int row1, int row2 )	Procedure yang akan menambahkan pertukaran baris pada proses OBE pada atribut stepByStep
addMkOnetoStep ( int row, double val )	Procedure yang akan menambahkan proses pembagian agar setiap baris memiliki 1 utama.
addNoSolutionsToStep ( )	Procedure yang menambahkan tulisan “Tidak ada Solusi” pada variable stepByStep
addGaussJordanRejected ( )	Procedure yang menambahkan tulisan bahwa proses Gauss Jordan ditolak pada variable stepByStep
addTitleStep ( )	Procedure yang menambahkan judul pada variable stepByStep
addGaussJordanAccepted ( )	Procedure yang menambahkan tulisan bahwa proses Gauss Jordan dilanjutkan pada variable stepByStep
addSolutionToStep ( )	Procedure yang akan
findIdxMain ( int row )	Fungsi yang akan menentukan indeks 1 utama setiap baris
refreshIdxMain ( )	Procedure yang akan menghitung ulang indeks 1 utama setiap baris setiap kali sudah dilakukan pengurangan.

subtractRow ( int row1, int row2 )	Procedure yang akan mengurangi baris row1 dengan baris row2 agar 1 utama baris row1 bergeser.
SwapRow ( int row1, int row2 )	Procedure yang digunakan untuk menukar posisi row1 dengan row2.
sortIdxMain ( int start, boolean wasSwapped )	Procedure yang akan mengurutkan posisi setiap baris menurut indeks 1 utama setiap baris mulai dari baris start hingga baris terakhir.
mkOneMain ( )	Procedure yang akan akan membagi setiap baris dengan elemen dengan indeks 1 utama pada setiap barisnya.
isSolusiUnik ( )	Fungsi yang akan menentukan apakah solusi dari persoalan yang sedang dihitung memiliki solusi unik atau tidak.
AddPatameterToStep ( )	Procedure yang akan menambahkan solusi parametrik ke dalam atribut stepByStep jika solusi persoalan tidak unik.
subtractJordan ( int row1, int row2, boolean isAdd )	Procedure yang akan mengurangi setiap elemen pada row1 dengan row2 untuk proses reverse subtraction pada OBE Gauss-Jordan.
obeGaussJordan ( )	Procedure yang akan melakukan OBE Gauss-Jordan.
gaussAndSolutions ( )	Procedure yang akan melakukan OBE Gauss.
setParameterSolutions ( )	Procedure yang akan menghitung solusi persoalan jika solusinya adalah solusi parametrik.

### c. Parameter

Class Parameter berisi atribut, konstruktor, fungsi , dan prosedur yang digunakan untuk mengatasi hasil parameter SPL yang sudah diproses.

Atribut	Deskripsi
Number	Atribut dengan tipe double yang menyimpan konstanta pada solusi parametrik
Var	Array of double yang menyimpan koefisien setiap variabel solusi parametrik
Terisi	Atribut dengan tipe Boolean mengindikasikan apakah pada sebuah solusi parametrik terdapat variabel (true) atau tidak (false).
Symbol	Atribut yang menyimpan char apa yang akan digunakan sebagai variabel parametrik.

#### • Konstruktor dan Selektor

Atribut	Deskripsi
Parameter ( )	Konstruktor obyek parameter.
SetParamtr ( double number, int idx, double koef, boolean terisi )	Procedure yang berfungsi untuk menset nilai setiap atribut parameter dengan input masukan.

#### • Fungsi dan Prosedur

Fungsi / Prosedur	Deskripsi
-------------------	-----------

isTerisi ()	Mengembalikan true jika parameter sudah pernah diisi atau false jika belum.
turnIntoString(int first, int last)	Mengubah setiap nilai yang terkandung di dalam parameter seperti number dan variable ke dalam tipe string.

#### d. Point

Class Point berisi atribut, konstruktor, fungsi, dan prosedur yang digunakan untuk membuat dan mengolah data di bidang kartesius yang nantinya dipakai untuk interpolasi

- **Atribut**

Atribut	Deskripsi
X	Variable yang bertipe double yang menampung data point di sumbu x
Y	Variable yang bertipe double yang menampung data point di sumbu y

- **Konstruktor**

Atribut	Deskripsi
Point ()	Konstruktor yang berfungsi untuk deklarasi tipe data Point
getX ()	Konstruktor ini untuk mengakses nilai data point di sumbu x
getY ()	Konstruktor ini untuk mengakses nilai data point di sumbu y
setX ()	Konstruktor ini untuk mengubah nilai data point di sumbu x
setY ()	Konstruktor ini untuk mengubah nilai data point di sumbu y

- **Fungsi dan Prosedure**

Fungsi / Prosedur	Deskripsi
readPoint ()	Procedure ini untuk menerima input point point dari user

## 2. Folder models

### a. BicubicSpline

- **Atribut**

Atribut	Deskripsi
initF	Menyimpan nilai dari 16 titik sumber (tipe Matriks)
invX	Menyimpan data invers X untuk proses Bicubic Spline Interpolation (tipe Matrix)
X	Menyimpan data X yang kemudian akan dihitung inversnya (tipe Matrix)
solveA	Menyimpan hasil perkalian invX
function	Bertipe String dan berfungsi menyimpan data $a_{ij}$ yang telah dihitung
Request	Array of double dan menyimpan nilai (x,y) yang akan dihitung

- **Konstruktor dan Selektor**

Konstruktor dan Selektor	Deskripsi
BicubicSpline( )	Mengonstruksi obyek BicubicSpline
setStaticInvX()	Menghitung Invers X

- **Fungsi dan Prosedure**

Fungsi / Prosedur	Deskripsi
inputInitFromText( )	Procedure yang akan menjalankan baca 16 titik <i>bicubic</i> dari file .txt
saveProcessesToText( )	Procedure yang akan menyimpan hasil perhitungan ke dalam file eksternal
solveBicubic( )	Procedure yang akan melakukan perhitungan untuk mendapatkan data $a_{ij}$ .
getRequestAnswer( )	Procedure yang akan menghitung $f(x,y)$
addFunctionToText( )	Procedure yang akan menambahkan isi $a_{ij}$ ke function untuk diprint

**b. DeterminanInvers**

- **Atribut**

Atribut	Deskripsi
contents	Tipe OBE yang menyimpan data Matrix
isDetZero	Tipe boolean sebagai indikasi Matrix tersebut memiliki determinan 0 atau tidak
sign	Setiap kali ada pertukaran baris saat reduksi sign akan berganti dari -1 ke 1 dan seterusnya.
multiply	Array of double yang akan menyimpan nilai saat ada suatu baris yang dikali atau dibagi.
countMul	Menyimpan berapa banyak isi multiply
result	Menyimpan hasil determinan setelah dihitung

- **Konstruktor**

Atribut	Deskripsi
DeterminanInvers( )	Konstruktor dengan jumlah baris dan kolom yang sudah di setel di bagian Matrix
DeterminanInvers(int row, int col)	Konstruktor dengan jumlah baris dan kolom langsung diset langsung

- **Fungsi dan Prosedure**

Fungsi / Prosedur	Deskripsi
inputMatriksFile( )	Procedure yang melakukan pembacaan isi Matrix lewat file text
saveToTextFile( )	Procedure yang menyimpan proses perhitungan ke dalam File
findImain(int row)	Fungsi yang akan mencari indeks 1 utama pada baris ke-row

refreshIMain( )	Menyimpan 1 utama dari setiap baris setelah dilakukan proses OBE.
swapforDet(int row1, int row2)	Procedure yang akan melakukan penukaran baris untuk operasi reduksi baris pada determinan
sortIMain(int start)	Procedure yang akan melakukan sorting menurut indeks 1 utama setiap baris mulai baris start.
isLanjut( )	Melakukan pengecekan terhadap proses perhitungan determinan dengan reduksi.
zeroDeterminan( )	Procedure yang akan menentukan apakah determinan 0 atau tidak
isLanjutForInvers( )	Fungsi yang akan mengecek keberlanjutan proses OBE untuk menghitung matriks balikan.
makeOneRow(int row)	Procedure yang melakukan pembagian agar setiap baris berelemen 1 pada indeks utamanya baris row.
makeOneAny(int start)	Procedure yang melakukan pembagian terhadap setiap baris mulai dari baris start agar pada indeks 1 utamanya berelemen 1
CalculateOBE( )	Procedure yang akan menghitung determinan matriks dengan metode reduksi.
determinanKofaktor( )	Procedure yang akan memanggil proses perhitungan determinan dari Matrix.class
addIdentity( )	Procedure yang akan menambahkan Matrix Identitas di belakang Matrix origin.
subtractJrdn(int row1, int row2)	Procedure yang akan melakukan proses substraksi pada baris row1 oleh baris row2
inversMatrix( )	Procedure yang akan melakukan proses perhitungan invers suatu Matrix dengan metode OBE.

### c. ImageBSI

#### • Atribut

Atribut	Deskripsi
DMat	Matrix yang menyimpan data D untuk <i>Image Processing</i>
XInvDmat	Matrix yang menyimpan hasil perkalian X invers dari <i>Bicubic Spline</i> dengan DMat
aValue	Menyimpan $a_{ij}$ hasil interpolasi.
sPath	<i>Source Path</i> sebagai asal file gambar
dPath	<i>Destination Path</i> sebagai tujuan file penyimpanan

#### • Konstruktor

Atribut	Deskripsi
ImageBSI ( )	Konstruktor overload
ImageBSI ( String sPath, String dPath)	Konstruktor class ImageBSI

#### • Fungsi dan Prosedur

Fungsi / Prosedur	Deskripsi
setDMat ( )	Procedure yang akan mengeset data dari Matrix DMat sesuai referensi
setXInvDmat ( )	Procedure yang akan mengeset data dari Matrix X invers dikali DMat

scaleImage ( double scale )	Procedure yang akan melakukan proses <i>Scalling Image</i> namun kompleksitas memorinya kurang baik. Saat ukuran gambar asal besar akan terjadi masalah karena memori penuh.
processImage ( double scale )	Procedure yang akan melakukan <i>Scalling Image</i> dengan kompleksitas memori dan waktu yang lebih baik.
get16Points (int i, int j BufferedImage input )	Fungsi yang akan mengeset matrix berisi nilai ARGB dari 16 titik yang adakan dilakukan interpolasi.
GetColorRGB ( int in )	Fungsi yang akan menghitung ARGB dari proses seleksi ARGB tiap pixel
getFValueOf ( double x, double y, Matrix [ ] a )	Fungsi yang akan menghitung ARGB yang baru sesuai proses interpolasi.

#### a. Interpolation

Class Interpolation berisi atribut, konstruktor, fungsi , dan prosedur yang digunakan untuk memecahkan permasalahan interpolasi dengan data-data input dari user.

##### • Atribut

Atribut	Deskripsi
Point	Variable yang bertipe matrix untuk menyimpan data titik titik bertipe double
xRequest	Variable yang bertipe double untuk menampung nilai x yang diminta user agar bisa menafsir nilai $f(x)$

##### • Konstruktor

Atribut	Deskripsi
Interpolation (int JmlhPoint)	Konstruktor untuk membuat matrix yang nantinya akan menyimpan data - data point. Matrix ini berdimensi 2 untuk kolom dan barisnya sebanyak parameter JmlhPoint

##### • Fungsi dan Prosedur

Fungsi / Prosedur	Deskripsi
ConvertPtoM (Matrix m)	Fungsi untuk mengolah dan mengonversi Matrix m yang berisikan data-data point menjadi Matrix SPL dan mengembalikan Matrix tersebut
askDataPoint ( )	Fungsi ini untuk menerima data-data point dari keyboard dan mengembalikan Matrix yang berisikan data – data point user
displayFunction ( OBE meselon)	Procedure yang menerima matrix SPL yang sudah menjadi matrix eselon dan menampilkan fungsi $f(x)$ interpolasi
taksiran ( OBE meselon, double input )	Fungsi ini mengolah variable input dan matrix meselon untuk mendapatkan hasil taksiran
askDataPointFromFile ( String path)	Procedure yang mengolah file path bertipe string lalu memasukkan data tersebut ke matrix dan menampung data yang diminta untuk ditafsir
uploadhasil2File (double taksiran, double xRequest, String filehasil)	Procedure ini mengolah hasil taksiran dan menuliskan hasil taksirannya ke file txt

#### b. Regresi

Class Regresi berisi atribut, konstruktor, fungsi, dan prosedur yang digunakan untuk memecahkan permasalahan Regresi dengan data-data input dari user.

- **Atribut**

Atribut	Deskripsi
regresiM	Variable bertipe matrix yang berfungsi untuk menampung hasil regresi
dataRegresiM	Variable bertipe matrix yang berfungsi menampung data data variable bebas dan terikat
listnilaivar	Variable bertipe matrix yang berfungsi menampung data konstanta variable bebas
jumlahPeubahValid	Variable bertipe integer yang berfungsi menampung banyak variable bebas dari persamaan regresi
banyakSampelValid	Variable bertipe integer yang berfungsi menampung banyak sampel user untuk diolah

- **Konstruktor dan Selektor**

Atribut	Deskripsi
Regresi ( int banyakSampel , int jumlahPeubah )	Konstruktor yang membuat matrix yang menampung data data regresi
getjumlahPeubah ( )	Konstruktor yang mengembalikan banyak variable bebas
getbanyakSampel ( )	Konstruktor yang mengembalikan banyak sampel

- **Fungsi dan Prosedure**

Fungsi / Prosedur	Deskripsi
askDataReg ( int sampel , int var )	Procedure ini menerima input data sampel sampel dari user dan memasukkan data tersebut ke dalam matrix dataRegresiM
convertData2Reg ( Matrix dataregresiM )	Procedure ini mengubah data – data regresi menjadi matrix regresi
convertReg2OBE ( Matrix regresiM )	Fungsi ini mengubah tipe data variable dataRegresiM menjadi tipe data OBE
displayFunction ( OBE meselon )	Procedure ini menampilkan fungsi regresi dari data regresi yang diberikan user
askDataRegFromFile ( String path )	Procedure ini menerima input data sampel sampel dari file txt dan memasukkan data tersebut ke dalam matrix dataRegresiM
Taksiran ( Matrix listnilaivar, OBE meselon )	Fungsi ini mengolah hasil regresi dan menghitung nilai taksiran dari input user
Uploadhasil2File ( double taksiran , String filehasil )	Procedure ini mengolah hasil taksiran dan menuliskan hasil taksirannya ke file txt

c. **SPL**

File ini berisikan blablabla

- **Atribut**

Atribut	Deskripsi
Spl	Variable ini untuk deklarasi tipe data OBE yang digunakan di file



Listnilaiavar	Variable ini untuk deklarasi tipe data Matrix yang digunakan untuk menampung solusi SPL
---------------	---

- **Konstruktor**

Atribut	Deskripsi
SPL ( )	Konstruktor ini untuk deklarasi tipe data SPL
SPL ( int row, int col )	Konstruktor ini untuk membuat tipe data SPL yang berdimensi kolom senilai variable col dan baris senilai variable row
getSPL ( )	Konstruktor ini untuk mengembalikan SPL baru

- **Fungsi dan Prosedure**

Fungsi / Prosedur	Deskripsi
inputSPLText ( )	Procedure ini untuk menerima input persamaan SPL lewat file txt dan mengolahnya ke dalam tipe data SPL
SaveToTextFile ( String path , String text )	Procedure ini untuk menyimpan hasil operasi ke dalam file txt
solveWithCramer (OBE mdata )	Procedure ini menyelesaikan SPL dengan menggunakan kaidah cramer
solveWithInverse ( )	Procedure ini menyelesaikan SPL dengan menggunakan metode invers
importToString ( Matrix m )	Fungsi ini mengembalikan solusi SPL dengan tipe data string

## BAB IV

### EKSPERIMEN

#### 4.1 Sistem Persamaan Linier

Proses Penyelesaian Persamaan Linier dapat dilakukan dengan 4 metode yang terlihat seperti ini:

```
===== Pilih Metode Penyelesaian SPL =====
1. Metode Eliminasi Gauss
2. Metode Eliminasi Gauss-Jordan
3. Metode Matriks Balikan
4. Kaidah Kramer
5. Kembali
>>> Pilih Metode :
```

##### a. Matriks Bersolusi Tunggal

- **Metode Eliminasi Gauss**

Dapat dilihat bahwa program berhasil menyelesaikan SPL dari matriks 3 x 4 yang bersolusi tunggal dengan metode eliminasi Gauss. Program juga memperlihatkan langkah-langkah untuk menemukan solusi tersebut.

```
>>> Pilih Metode : 1

===== Pilih Metode Masukkan =====
1. Masukkan Ketik
2. Masukkan dalam bentuk File
3. Kembali
>>> Pilih Metode Masukkan : 1

===== Metode Ketik =====
Ketik -1 untuk kembali
>>> Masukkan Ukuran Matriks baris x kolom (m x n) : 3 4
>>> Masukkan Isi Matriks 3 x 4 :
2 1 4 4
3 2 0 1
1 1 2 3

===== HASIL =====

      2.0      1.0      4.0      4.0
      3.0      2.0      0.0      1.0
      1.0      1.0      2.0      3.0

<><><><> Langkah Penyelesaian <><><><>

>>> Matriks Augmented Awal:

      2.0      1.0      4.0      4.0
      3.0      2.0      0.0      1.0
      1.0      1.0      2.0      3.0

>>> 2R2 - (3)R1
>>> 2R3 - R1

      2.0      1.0      4.0      4.0
      0.0      1.0     -12.0     -10.0
      0.0      1.0      0.0      2.0

>>> R3 - R2

      2.0      1.0      4.0      4.0
      0.0      1.0     -12.0     -10.0
      0.0      0.0      12.0      12.0

>>> R1/2
>>> R3/12

      1.0      0.5      2.0      2.0
      0.0      1.0     -12.0     -10.0
      0.0      0.0      1.0      1.0

----> X1 = -1.0
----> X2 = 2.0
----> X3 = 1.0
```

- **Metode Eliminasi Gauss-Jordan**

Dapat dilihat bahwa program berhasil menyelesaikan SPL dari matriks 3 x 4 yang bersolusi tunggal dengan metode eliminasi Gauss-Jordan atau melanjutkan hasil OBE pada Metode Eliminasi Gauss. Program juga memperlihatkan langkah-langkah untuk menemukan solusi tersebut.

```

>>> Pilih Metode : 2

===== Pilih Metode Masukkan =====
1. Masukkan Ketik
2. Masukkan dalam bentuk File
3. Kembali
>>> Pilih Metode Masukkan : 1

===== Metode Ketik =====
Ketik -1 untuk kembali
>>> Masukkan Ukuran Matriks baris x kolom (m x n) : 3 4
>>> Masukkan Isi Matriks 3 x 4 :
2 1 4 4
3 2 0 1
1 1 2 3

===== HASIL =====

      2.0      1.0      4.0      4.0
      3.0      2.0      0.0      1.0
      1.0      1.0      2.0      3.0

<<<<<<< Langkah Penyelesaian <<<<<<<

>>> Matriks Augmented Awal:

      2.0      1.0      4.0      4.0
      3.0      2.0      0.0      1.0
      1.0      1.0      2.0      3.0

>>> 2R2 - (3)R1
>>> 2R3 - R1

      2.0      1.0      4.0      4.0
      0.0      1.0     -12.0     -10.0
      0.0      1.0      0.0      2.0

|
>>> R3 - R2

      2.0      1.0      4.0      4.0
      0.0      1.0     -12.0     -10.0
      0.0      0.0      12.0      12.0

>>> R1/2
>>> R3/12

      1.0      0.5      2.0      2.0
      0.0      1.0     -12.0     -10.0
      0.0      0.0      1.0      1.0

```

>>> Lanjutan Proses Gauss-Jordan

```

>>> R2 - (-12)R3
>>> R1 - (2)R3
>>> 2R1 - R2

      1.0      0.0      0.0     -1.0
      0.0      1.0      0.0      2.0
      0.0      0.0      1.0      1.0

      1.0      0.0      0.0     -1.0
      0.0      1.0      0.0      2.0
      0.0      0.0      1.0      1.0

```

```

----> X1 = -1.0
----> X2 = 2.0
----> X3 = 1.0

```

## • Metode Matriks Balikan

Program dapat menyelesaikan Sistem Persamaan Linier tersebut dengan menggunakan metode matriks balikan, karena ukuran matriks yang persegi ( $Ax = B$ , dengan A harus persegi dan memiliki inverse). Hasil dapat dilihat seperti dibawah.

```

>>> Pilih Metode : 3

===== Pilih Metode Masukkan =====
1. Masukkan Ketik
2. Masukkan dalam bentuk File
3. Kembali
>>> Pilih Metode Masukkan : 1

===== Metode Ketik =====
Ketik -1 untuk kembali
>>> Masukkan Ukuran Matriks (n x (n+1)), n : 3
>>> Masukkan Isi Matriks 3 x 4 :
2 1 4 4
3 2 0 1
1 1 2 3

```

```

===== HASIL =====

      2.0      1.0      4.0      4.0
      3.0      2.0      0.0      1.0
      1.0      1.0      2.0      3.0

===== Invers X =====

0.5 0.5 -2.5
0.0 1.0 -3.0
0.0 0.0 1.0

----> X1 = -5.0
----> X2 = -8.0
----> X3 = 3.0

```

- **Kaidah Kramer**

Program dapat menyelesaikan Sistem Persamaan Linier tersebut dengan Kaidah Kramer, karena ukuran matriks yang persegi ( $Ax = B$ , dengan A harus persegi dan memiliki inverse). Hasil dapat dilihat seperti dibawah

```
>>> Pilih Metode : 4

===== Pilih Metode Masukkan =====
1. Masukkan Ketik
2. Masukkan dalam bentuk File
3. Kembali
>>> Pilih Metode Masukkan : 1

===== Metode Ketik =====
Ketik -1 untuk kembali
>>> Masukkan Ukuran Matriks (n x (n+1)), n : 3
>>> Masukkan Isi Matriks 3 x 4 :
2 1 4 4
3 2 0 1
1 1 2 3
===== HASIL =====

      2.0      1.0      4.0      4.0
      3.0      2.0      0.0      1.0
      1.0      1.0      2.0      3.0

Solusi :
--> X1 = -1.0
--> X2 = 2.0
--> X3 = 1.0
```

## b. Matriks Tidak Bersolusi

- **Metode Eliminasi Gauss dan Gauss Jordan**

```
>>> Pilih menu : 1

===== Pilih Metode Penyelesaian SPL =====
1. Metode Eliminasi Gauss
2. Metode Eliminasi Gauss-Jordan
3. Metode Matriks Balikan
4. Kaidah Kramer
5. Kembali
>>> Pilih Metode : 1

===== Pilih Metode Masukkan =====
1. Masukkan Ketik
2. Masukkan dalam bentuk File
3. Kembali
>>> Pilih Metode Masukkan : 2
Ketik 0 untuk kembali
>>> Masukkan nama file: test2.txt

===== HASIL =====
<><><> Langkah Penyelesaian <><><>
....
1.0      1.0      1.0      0.0      0.0      0.0      0.0      0.0      8.0
0.0 0.1360400 -0.57107      0.75      0.04289      0.0      0.04289      0.0      0.0 -1.101679
0.0      0.0      -0.70711      0.88604      0.04289      0.0      0.1789300      0.0      0.0 -1.373759
0.0      0.0      0.0      -0.051622 -0.003332 -0.072146 -0.064709      0.0      0.0 -0.867388
0.0      0.0      0.0      0.0      0.0046977 0.0062902 9.1577656 0.0012414 0.0045398 0.1134885
0.0      0.0      0.0      0.0      0.0      -2.296785 -7.342711 1.7513138 1.0625730 -1.173886
0.0      0.0      0.0      0.0      0.0      0.0      -6.289665 -2.216127 5.1779708 1.1574054
0.0      0.0      0.0      0.0      0.0      0.0      0.0      -7.106580 -2.206660 -5.380446
0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0      -7.550573 2.6325596
0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0      1.4107737
0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0      1.2270810
0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0      -2.561247

>>> R2/(0.1360400)
>>> R3/(-0.70711)
>>> R4/(-0.051622)
>>> R5/(0.0046977)
>>> R6/(-2.296785)
>>> R7/(-6.289665)
>>> R8/(-7.106580)
>>> R9/(-7.550573)
```

Tidak ada Solusi

- **Metode Matriks Balikan dan Kaidah Kramer**

Pada proses penyelesaian Matriks Balikan dan Kaidah Kramer, dilakukan pengecekan nilai determinan. Suatu matriks yang determinannya bernilai 0, tidak dapat diselesaikan.

```
>>> Pilih menu : 1

===== Pilih Metode Penyelesaian SPL =====
1. Metode Eliminasi Gauss
2. Metode Eliminasi Gauss-Jordan
3. Metode Matriks Balikan
4. Kaidah Kramer
5. Kembali
>>> Pilih Metode : 4

===== Pilih Metode Masukkan =====
1. Masukkan Ketik
2. Masukkan dalam bentuk File
3. Kembali
>>> Pilih Metode Masukkan : 2
Ketik 0 untuk kembali
>>> Masukkan nama file: test2.txt
===== HASIL =====
Tidak dapat menghitung solusi karena matriks tidak memiliki Invers.
```

### c. Matriks Bersolusi Parameter

- **Metode Eliminasi Gauss dan Gauss-Jordan**

```
>>> Pilih Metode : 2

===== Pilih Metode Masukkan =====
1. Masukkan Ketik
2. Masukkan dalam bentuk File
3. Kembali
>>> Pilih Metode Masukkan : 2
Ketik 0 untuk kembali
>>> Masukkan nama file: test3.txt
===== HASIL =====
<><><> Langkah Penyelesaian <><><>

>>> Matriks Augmented Awal:

    1.0   -1.0   0.0   0.0   1.0   3.0
    1.0    1.0   0.0  -3.0   0.0   6.0
    2.0   -1.0   0.0   1.0  -1.0   5.0
    -1.0    2.0   0.0  -2.0  -1.0  -1.0

.....
.....
.....
    1.0   -1.0   0.0   0.0   1.0   3.0
    0.0    2.0   0.0  -3.0  -1.0   3.0
    0.0    0.0   0.0   5.0  -5.0  -5.0
    0.0    0.0   0.0   0.0   0.0   0.0

>>> R2/2
>>> R3/5

Tidak dapat melanjutkan proses Gauss-Jordan karena solusi tidak unik.

    1.0   -1.0   0.0   0.0   1.0   3.0
    0.0    1.0   0.0  -1.5  -0.5   1.5
    0.0    0.0   0.0   1.0  -1.0  -1.0
    0.0    0.0   0.0   0.0   0.0   0.0

---> X1 = 3 + φ5
---> X2 = 2φ5
---> X3 = φ3
---> X4 = -1 + φ5
---> X5 = φ5
```

Dengan metode Gauss-Jordan dan Gauss akan menghasilkan solusi yang sama. Dapat dilihat bahwa solusi dari matriks adalah dalam bentuk parameter sehingga dikeluarkan dengan  $\phi$  melambangkan parameter. Dengan  $\phi_n$  adalah parameter ke- $n$ .

- **Metode Matriks Balikan dan Kaidah Kramer**

Pada proses penyelesaian Matriks Balikan dan Kaidah Kramer, dilakukan pengecekan nilai determinan serta adakah tidak invers dari matriks. Karena pada matriks tidak memiliki inverse maka tidak dapat diselesaikan menggunakan metode ini.

```

>>> Pilih Metode : 3

===== Pilih Metode Masukkan =====
1. Masukkan Ketik
2. Masukkan dalam bentuk File
3. Kembali
>>> Pilih Metode Masukkan : 2
Ketik 0 untuk kembali
>>> Masukkan nama file: test3.txt
===== HASIL =====
Tidak dapat menghitung solusi karena matriks tidak memiliki Invers.

```

#### d. Invalid

Karena input dan output dari SPL adalah satu block kode yang sama maka hanya perlu menunjukkan satu kali saja untuk semua Metode.

- **Salah ukuran**

```

>>> Pilih Metode : 1

===== Pilih Metode Masukkan =====
1. Masukkan Ketik
2. Masukkan dalam bentuk File
3. Kembali
>>> Pilih Metode Masukkan : 1

===== Metode Ketik =====
Ketik -1 untuk kembali
>>> Masukkan Ukuran Matriks baris x kolom (m x n) : a
Ukuran Matriks harus bilangan bulat positif

```

- **Salah isi**

```

===== Metode Ketik =====
Ketik -1 untuk kembali
>>> Masukkan Ukuran Matriks baris x kolom (m x n) : 4 3
>>> Masukkan Isi Matriks 4 x 3 :
1 2 3 4
4 3 2 a
Isi Matrix salah. Pastikan isi matriks adalah bilangan real

```

## 4.2 Determinan

### a. Metode Ekspansi Kofaktor

Dapat dilihat bahwa program berhasil menentukan hasil determinan dari matriks persegi 4x4. Hasil dapat dilihat dibawah ini.

```

>>> Pilih menu : 2

===== Pilih Metode Determinan =====
1. Metode Ekspansi Kofaktor
2. Metode Reduksi Baris dengan OBE
3. Kembali
>>> Pilih Metode Determinan : 1

===== Pilih Metode Masukkan =====
1. Masukkan Ketik
2. Masukkan dalam bentuk File
3. Kembali
>>> Pilih Metode Masukkan : 1

===== Metode Ketik =====
Ketik -1 untuk kembali
>>> Masukkan Ukuran Matriks (n x n) : 4
>>> Masukkan Isi Matriks 4 x 4 :
8 1 3 2
2 9 -1 -2
1 3 2 -1
1 0 6 4

===== HASIL =====
Determinan = 740.0

```

## b. Metode Reduksi Baris dengan OBE

Dapat dilihat dengan input dari file txt, operasi Determinan dapat berjalan dan menghasilkan output yang benar. ( isi file dan input ketik sebelumnya sama)

```

>>> Pilih menu : 2

===== Pilih Metode Determinan =====
1. Metode Ekspansi Kofaktor
2. Metode Reduksi Baris dengan OBE
3. Kembali
>>> Pilih Metode Determinan : 2

===== Pilih Metode Masukkan =====
1. Masukkan Ketik
2. Masukkan dalam bentuk File
3. Kembali
>>> Pilih Metode Masukkan : 2
Ketik 0 untuk kembali
>>> Masukkan nama file: test1.txt
===== HASIL =====
<><><><> Langkah Penyelesaian <><><><>

>>> Matriks Augmented Awal:

      8.0    1.0    3.0    2.0
      2.0    9.0   -1.0   -2.0
      1.0    3.0    2.0   -1.0
      1.0    0.0    6.0    4.0

>>> R1/8

      1.0    0.125    0.375    0.25
      2.0    9.0   -1.0   -2.0
      1.0    3.0    2.0   -1.0
      1.0    0.0    6.0    4.0

.....
.....
.....

>>> R4 - R3

      1.0    0.125    0.375    0.25
      0.0    1.0   -0.2  -0.2857
      0.0    0.0    1.0  -0.1948
      0.0    0.0    0.0  0.85807

>>> R4/(0.8580705)

      1.0    0.125    0.375    0.25
      0.0    1.0   -0.2  -0.2857
      0.0    0.0    1.0  -0.1948
      0.0    0.0    0.0    1.0

Determinan = 740.0000000000001

```

## 4.3 Matriks Balikan

### a. Matriks Balikan

Program dapat menentukan matriks balikan dengan input ketik.

```
>>> Pilih menu : 3

===== Pilih Metode Matriks Balikan =====
1. Metode Matriks Balikan
2. Metode Adjoin
3. Kembali
>>> Pilih Metode Determinan : 1

===== Pilih Metode Masukkan =====
1. Masukkan Ketik
2. Masukkan dalam bentuk File
3. Kembali
>>> Pilih Metode Masukkan : 1

===== Metode Ketik =====
Ketik -1 Kembali
>>> Masukkan Ukuran Matriks (n x n) : 4
>>> Masukkan Isi Matriks 4 x 4 :
8 1 3 2
2 9 -1 -2
1 3 2 -1
1 0 6 4
===== HASIL =====
Hasil Inverse :
0.13783    -0.0189    0.01081    -0.0756
-0.0337    0.14189   -0.0810    0.06756
-0.0202    -0.1148    0.35135    0.04054
-0.0040    0.17702   -0.5297    0.20810
```

### b. Metode Adjoin

Program dapat menentukan matriks balikan dengan metode Adjoin dan Kofaktor serta Determinan. Program juga dapat menerima input dari file ( isi file dan input ketik sebelumnya sama)

```
>>> Pilih menu : 3

===== Pilih Metode Matriks Balikan =====
1. Metode Matriks Balikan
2. Metode Adjoin
3. Kembali
>>> Pilih Metode Determinan : 2

===== Pilih Metode Masukkan =====
1. Masukkan Ketik
2. Masukkan dalam bentuk File
3. Kembali
>>> Pilih Metode Masukkan : 2
Ketik 0 untuk kembali
>>> Masukkan nama file: test1.txt
===== HASIL =====
Hasil Inverse :
0.13783    -0.0189    0.01081    -0.0756
-0.0337    0.14189   -0.0810    0.06756
-0.0202    -0.1148    0.35135    0.04054
-0.0040    0.17702   -0.5297    0.20810
```



## 4.4 Interpolasi Polinom

- a. Gunakan tabel di bawah ini untuk mencari polinom interpolasi dari pasangan titik-titik yang terdapat dalam tabel. Program menerima masukan nilai  $x$  yang akan dicari nilai fungsi  $f(x)$ .

$x$	0.1	0.3	0.5	0.7	0.9	1.1	1.3
$f(x)$	0.003	0.067	0.148	0.248	0.370	0.518	0.697

Pengujian pada nilai  $x = 0.2$  :

```

**** APLIKASI MULAI ****
Selamat Datang pada Aplikasi Perhitungan Matriks dan SPL

===== MENU UTAMA =====
1. Sistem Persamaan Linier
2. Determinan
3. Matriks balikan
4. Interpolasi Polinom
5. Interpolasi Bicubic Spline
6. Regresi linier berganda
7. Pemecahan Gambar dengan Interpolasi Bicubic Spline
8. Keluar
>>> Pilih menu : 4

===== Pilih Metode Masukan =====
1. Masukkan Ketik
2. Masukkan dalam bentuk file
3. Kembali
>>> Pilih Metode Masukan : 2
Ketik 0 untuk kembali
>>> Masukkan nama file: polinom.txt

===== HASIL =====
Fungsi : f(x) = 0.013x^6 + 0.005800000000000001x^5 + 0.013000000000000001x^4 + 0.0227x^3 + 0.0341x^2 + 0.048500000000000001x + 0.0658
Taksiran f(0.2) = 0.0755
  
```

Pengujian pada nilai  $x = 0.55$  :

```
2. Determinan
3. Matriks balikan
4. Interpolasi Polinom
5. Interpolasi Bicubic Spline
6. Regresi linier berganda
7. Pembesaran Gambar dengan Interpolasi Bicubic Spline
8. Keluar
>>> Pilih menu : 4

----- Pilih Metode Masukkan -----
1. Masukkan Ketik
2. Masukkan dalam bentuk File
3. Kembali
>>> Pilih Metode Masukkan : 2
Ketik 0 untuk kembali
>>> Masukkan nama file: polinom.txt

----- HASIL -----
Fungsi :  $f(x) = 0.63x^5 + 0.005800000000000001x^4 + 0.013300000000000001x^3 + 0.0227x^2 + 0.0343x + 0.048500000000000001$ 
Taksiran  $f(0.2) = 0.0755$ 

>>> Apakah ingin disimpan dalam file? [y/n]: n

----- MENU UTAMA -----
1. Sistem Persamaan Linier
2. Determinan
3. Matriks balikan
4. Interpolasi Polinom
5. Interpolasi Bicubic Spline
6. Regresi linier berganda
7. Pembesaran Gambar dengan Interpolasi Bicubic Spline
8. Keluar
>>> Pilih menu : 4

----- Pilih Metode Masukkan -----
1. Masukkan Ketik
2. Masukkan dalam bentuk File
3. Kembali
>>> Pilih Metode Masukkan : 2
Ketik 0 untuk kembali
>>> Masukkan nama file: polinom.txt

----- HASIL -----
Fungsi :  $f(x) = 0.63x^5 + 0.005800000000000001x^4 + 0.013300000000000001x^3 + 0.0227x^2 + 0.0343x + 0.048500000000000001$ 
Taksiran  $f(0.55) = 0.092475$ 

>>> Apakah ingin disimpan dalam file? [y/n]:
```

Pengujian pada nilai  $x = 0.85$  :

```
2. Determinan
3. Matriks balikan
4. Interpolasi Polinom
5. Interpolasi Bicubic Spline
6. Regresi linier berganda
7. Pembesaran Gambar dengan Interpolasi Bicubic Spline
8. Keluar
>>> Pilih menu : 4

----- Pilih Metode Masukkan -----
1. Masukkan Ketik
2. Masukkan dalam bentuk File
3. Kembali
>>> Pilih Metode Masukkan : 2
Ketik 0 untuk kembali
>>> Masukkan nama file: polinom.txt

----- HASIL -----
Fungsi :  $f(x) = 0.63x^5 + 0.005800000000000001x^4 + 0.013300000000000001x^3 + 0.0227x^2 + 0.0343x + 0.048500000000000001$ 
Taksiran  $f(0.55) = 0.092475$ 

>>> Apakah ingin disimpan dalam file? [y/n]: n

----- MENU UTAMA -----
1. Sistem Persamaan Linier
2. Determinan
3. Matriks balikan
4. Interpolasi Polinom
5. Interpolasi Bicubic Spline
6. Regresi linier berganda
7. Pembesaran Gambar dengan Interpolasi Bicubic Spline
8. Keluar
>>> Pilih menu : 4

----- Pilih Metode Masukkan -----
1. Masukkan Ketik
2. Masukkan dalam bentuk File
3. Kembali
>>> Pilih Metode Masukkan : 2
Ketik 0 untuk kembali
>>> Masukkan nama file: polinom.txt

----- HASIL -----
Fungsi :  $f(x) = 0.63x^5 + 0.005800000000000001x^4 + 0.013300000000000001x^3 + 0.0227x^2 + 0.0343x + 0.048500000000000001$ 
Taksiran  $f(0.55) = 0.107025000000000001$ 

>>> Apakah ingin disimpan dalam file? [y/n]:
```

Pengujian pada nilai  $x = 1.28$  :

```

2. Determinan
3. Matriks balikan
4. Interpolasi Polinom
5. Interpolasi Bicubic Spline
6. Regresi linier berganda
7. Pembesaran Gambar dengan Interpolasi Bicubic Spline
8. Keluar
>>> Pilih menu : 4

----- Pilih Metode Masukan -----
1. Masukkan Ketik
2. Masukkan dalam bentuk File
3. Kembali
>>> Pilih Metode Masukan : 2
Ketik 0 untuk kembali
>>> Masukkan nama file: polinom.txt

HASIL -----
Fungsi : f(X) = 0.63X^6 + 0.005800000000000001X^5 + 0.013300000000000001X^4 + 0.0227X^3 + 0.0343X^2 + 0.048500000000000001X + 0.0658
Taksiran f(0.85) = 0.107025000000000001

>>> Apakah ingin disimpan dalam file? [y/n]: n

----- MENU UTAMA -----
1. Sistem Persamaan Linier
2. Determinan
3. Matriks balikan
4. Interpolasi Polinom
5. Interpolasi Bicubic Spline
6. Regresi linier berganda
7. Pembesaran Gambar dengan Interpolasi Bicubic Spline
8. Keluar
>>> Pilih menu : 4

----- Pilih Metode Masukan -----
1. Masukkan Ketik
2. Masukkan dalam bentuk File
3. Kembali
>>> Pilih Metode Masukan : 2
Ketik 0 untuk kembali
>>> Masukkan nama file: polinom.txt

HASIL -----
Fungsi : f(X) = 0.63X^6 + 0.005800000000000001X^5 + 0.013300000000000001X^4 + 0.0227X^3 + 0.0343X^2 + 0.048500000000000001X + 0.0658
Taksiran f(1.28) = 0.263688000000000003

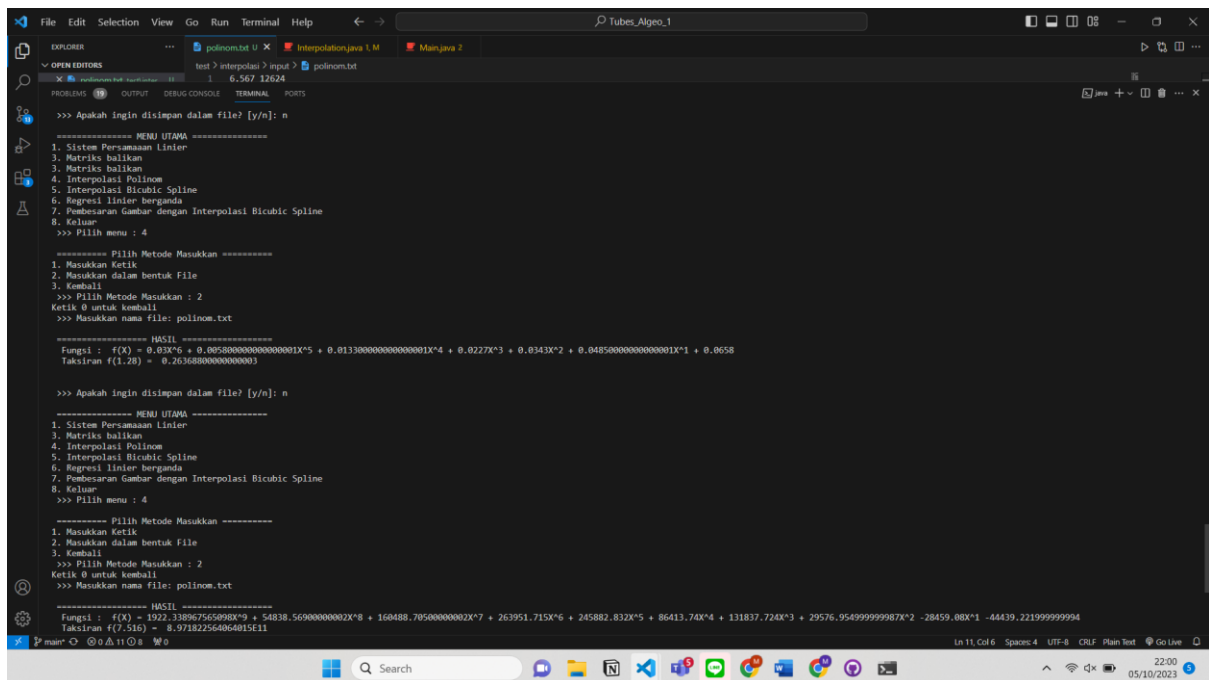
>>> Apakah ingin disimpan dalam file? [y/n]:

```

b. Jumlah kasus positif baru Covid-19 di Indonesia semakin fluktuatif dari hari ke hari. Di bawah ini diperlihatkan jumlah kasus baru Covid-19 di Indonesia mulai dari tanggal 17 Juni 2022 hingga 31 Agustus 2022:

Tanggal	Tanggal (desimal)	Jumlah Kasus Baru
17/06/2022	6,567	12.624
30/06/2022	7	21.807
08/07/2022	7,258	38.391
14/07/2022	7,451	54.517
17/07/2022	7,548	51.952
26/07/2022	7,839	28.228
05/08/2022	8,161	35.764
15/08/2022	8,484	20.813
22/08/2022	8,709	12.408
31/08/2022	9	10.534

Pengujian pada tanggal 16/07/2022 :



```
>>> Apakah ingin disimpan dalam file? [y/n]: n

===== MENU UTAMA =====
1. Sistem Persamaan Linier
2. Matriks balikan
3. Matriks balikan
4. Interpolasi Polinom
5. Interpolasi Bicubic Spline
6. Regresi linier berganda
7. Pembesaran Gambar dengan Interpolasi Bicubic Spline
8. Keluar
>>> Pilih menu : 4

===== Pilih Metode Masukkan =====
1. Masukkan Ketik
2. Masukkan dalam bentuk File
3. Kembali
>>> Pilih Metode Masukkan : 2
Ketik 0 untuk kembali
>>> Masukkan nama file: polinom.txt

===== HASIL =====
Fungsi :  $f(x) = 0.01x^6 + 0.005000000000000001x^5 + 0.011000000000000001x^4 + 0.0227x^3 + 0.0341x^2 + 0.040500000000000001x + 0.0650$ 
Taksiran  $f(1.28) = 0.2636800000000000$ 

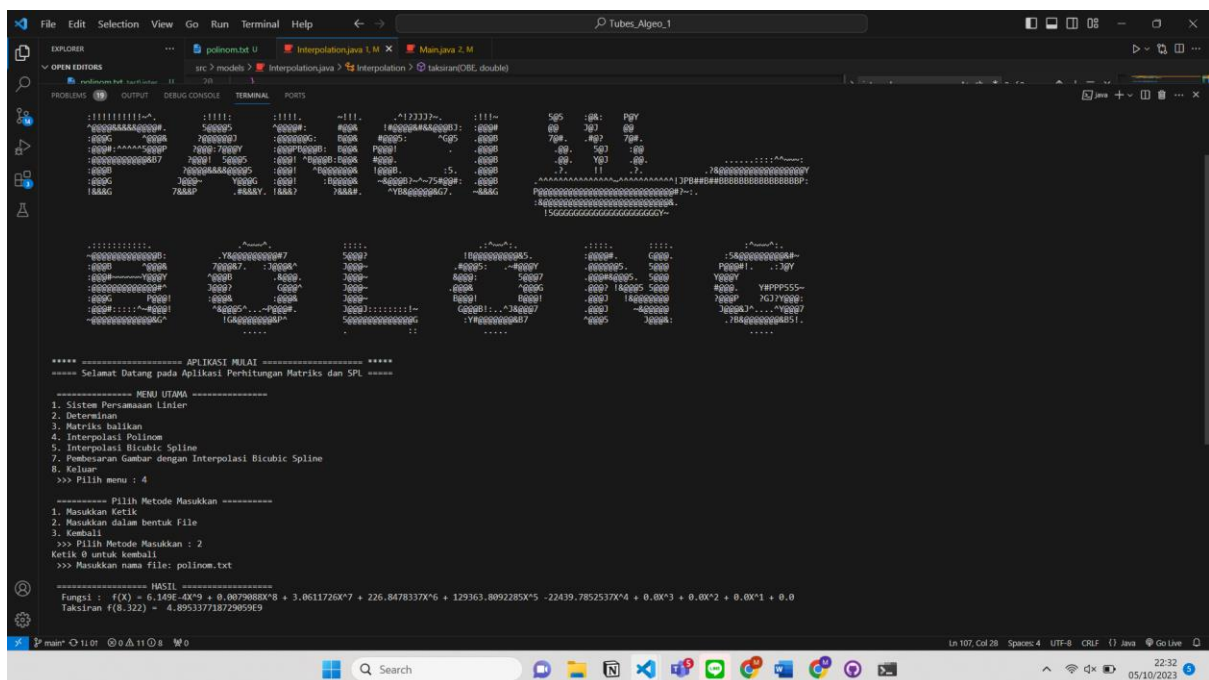
>>> Apakah ingin disimpan dalam file? [y/n]: n

===== MENU UTAMA =====
1. Sistem Persamaan Linier
2. Matriks balikan
3. Matriks balikan
4. Interpolasi Polinom
5. Interpolasi Bicubic Spline
6. Regresi linier berganda
7. Pembesaran Gambar dengan Interpolasi Bicubic Spline
8. Keluar
>>> Pilih menu : 4

===== Pilih Metode Masukkan =====
1. Masukkan Ketik
2. Masukkan dalam bentuk File
3. Kembali
>>> Pilih Metode Masukkan : 2
Ketik 0 untuk kembali
>>> Masukkan nama file: polinom.txt

===== HASIL =====
Fungsi :  $f(x) = 1922.3896756908x^9 + 54838.569000000002x^8 + 160488.70500000002x^7 + 263951.715x^6 + 245882.832x^5 + 86413.74x^4 + 131837.724x^3 + 29576.954999999987x^2 - 28459.08x - 44439.221999999994$ 
Taksiran  $f(7.516) = 8.97182564064015E11$ 
```

Pengujian pada tanggal 10/08/2022 :



```
***** APLIKASI MULAI *****
**** Selamat Datang pada Aplikasi Perhitungan Matriks dan SPL ****

===== MENU UTAMA =====
1. Sistem Persamaan Linier
2. Determinan
3. Matriks balikan
4. Interpolasi Polinom
5. Interpolasi Bicubic Spline
6. Regresi linier berganda
7. Pembesaran Gambar dengan Interpolasi Bicubic Spline
8. Keluar
>>> Pilih menu : 4

===== Pilih Metode Masukkan =====
1. Masukkan Ketik
2. Masukkan dalam bentuk File
3. Kembali
>>> Pilih Metode Masukkan : 2
Ketik 0 untuk kembali
>>> Masukkan nama file: polinom.txt

===== HASIL =====
Fungsi :  $f(x) = 6.149E-4x^9 + 0.00790808x^8 + 3.0611726x^7 + 226.8478337x^6 + 129363.8092285x^5 - 22439.7852537x^4 + 0.0x^3 + 0.0x^2 + 0.0x^1 + 0.0$ 
Taksiran  $f(0.322) = 4.895337710729059E9$ 
```

Pengujian pada tanggal 05/09/2022 :

```

===== MENU UTAMA =====
1. Sistem Persamaan Linier
2. Determinan
3. Matriks balikan
4. Interpolasi Polinom
5. Interpolasi Bicubic Spline
6. Regresi linier berganda
7. Pembesaran Gambar dengan Interpolasi Bicubic Spline
8. Keluar
>>> Pilih menu : 4

===== Pilih Metode Masukkan =====
1. Masukkan Ketik
2. Masukkan dalam bentuk File
3. Kembali
>>> Pilih Metode Masukkan : 2
Ketik 0 untuk kembali
>>> Masukkan nama file: polinom.txt

===== HASIL =====
Fungsi : f(X) = 6.149E-4X^9 + 0.0079088X^8 + 3.0611726X^7 + 226.8478337X^6 + 129363.8092285X^5 - 22439.7852537X^4 + 0.0X^3 + 0.0X^2 + 0.0X^1 + 0.0
Taksiran f(8.322) = 4.89533718729059E9

>>> Apakah ingin disimpan dalam file? [y/n]: n

===== MENU UTAMA =====
1. Sistem Persamaan Linier
2. Determinan
3. Matriks balikan
4. Interpolasi Polinom
5. Interpolasi Bicubic Spline
6. Regresi linier berganda
7. Pembesaran Gambar dengan Interpolasi Bicubic Spline
8. Keluar
>>> Pilih menu : 4

===== Pilih Metode Masukkan =====
1. Masukkan Ketik
2. Masukkan dalam bentuk File
3. Kembali
>>> Pilih Metode Masukkan : 2
Ketik 0 untuk kembali
>>> Masukkan nama file: polinom.txt

===== HASIL =====
Fungsi : f(X) = 6.149E-4X^9 + 0.0079088X^8 + 3.0611726X^7 + 226.8478337X^6 + 129363.8092285X^5 - 22439.7852537X^4 + 0.0X^3 + 0.0X^2 + 0.0X^1 + 0.0
Taksiran f(5.107) = 8.157219856394142E9

```

Pengujian pada tanggal 16/04/2022:

```

===== MENU UTAMA =====
1. Sistem Persamaan Linier
2. Determinan
3. Matriks balikan
4. Interpolasi Polinom
5. Interpolasi Bicubic Spline
6. Regresi linier berganda
7. Pembesaran Gambar dengan Interpolasi Bicubic Spline
8. Keluar
>>> Pilih menu : 4

===== Pilih Metode Masukkan =====
1. Masukkan Ketik
2. Masukkan dalam bentuk File
3. Kembali
>>> Pilih Metode Masukkan : 2
Ketik 0 untuk kembali
>>> Masukkan nama file: polinom.txt

===== HASIL =====
Fungsi : f(X) = 6.149E-4X^9 + 0.0079088X^8 + 3.0611726X^7 + 226.8478337X^6 + 129363.8092285X^5 - 22439.7852537X^4 + 0.0X^3 + 0.0X^2 + 0.0X^1 + 0.0
Taksiran f(9.107) = 8.157219856394142E9

>>> Apakah ingin disimpan dalam file? [y/n]: n

===== MENU UTAMA =====
1. Sistem Persamaan Linier
2. Determinan
3. Matriks balikan
4. Interpolasi Polinom
5. Interpolasi Bicubic Spline
6. Regresi linier berganda
7. Pembesaran Gambar dengan Interpolasi Bicubic Spline
8. Keluar
>>> Pilih menu : 4

===== Pilih Metode Masukkan =====
1. Masukkan Ketik
2. Masukkan dalam bentuk File
3. Kembali
>>> Pilih Metode Masukkan : 2
Ketik 0 untuk kembali
>>> Masukkan nama file: polinom.txt

===== HASIL =====
Fungsi : f(X) = 6.149E-4X^9 + 0.0079088X^8 + 3.0611726X^7 + 226.8478337X^6 + 129363.8092285X^5 - 22439.7852537X^4 + 0.0X^3 + 0.0X^2 + 0.0X^1 + 0.0
Taksiran f(4.533) = 2.0878809392853504E8

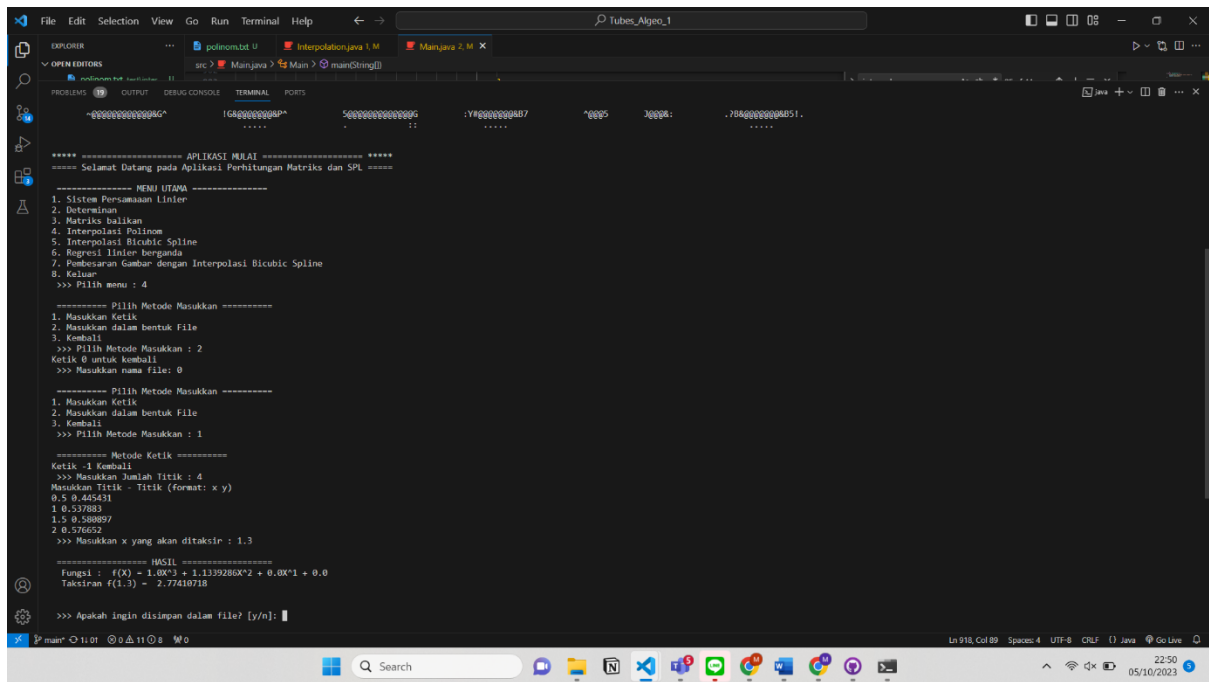
```

a. C. Sederhanakan fungsi  $f(x)$  yang memenuhi kondisi

$$f(x) = \frac{x^2 + \sqrt{x}}{e^x + x}$$

dengan polinom interpolasi derajat  $n$  di dalam selang  $[0, 2]$ .

Sebagai contoh, jika  $n = 5$ , maka titik-titik  $x$  yang diambil di dalam selang  $[0, 2]$  berjarak  $h = (2 - 0)/5 = 0.4$ .



```
===== APLIKASI MULAI =====
----- Selamat Datang pada Aplikasi Perhitungan Matriks dan SPL -----

MENU UTAMA
1. Sistem Persamaan Linier
2. Determinan
3. Matriks balikan
4. Interpolasi Polinom
5. Interpolasi Bicubic Spline
6. Regresi linier berganda
7. Pembesaran Gambar dengan Interpolasi Bicubic Spline
8. Keluar
>>> Pilih menu : 4

===== Pilih Metode Masukkan =====
1. Masukkan Ketik
2. Masukkan dalam bentuk File
3. Kembali
>>> Pilih Metode Masukkan : 2
Ketik 0 untuk kembali
>>> Masukkan nama file: 0

===== Pilih Metode Masukkan =====
1. Masukkan Ketik
2. Masukkan dalam bentuk File
3. Kembali
>>> Pilih Metode Masukkan : 1

===== Metode Ketik =====
Ketik -1 Kembali
>>> Masukkan Jumlah Titik : 4
Masukkan Titik - Titik (format: x y)
0.5 0.445431
1 0.537883
1.5 0.589657
2 0.576652
>>> Masukkan x yang akan ditaksir : 1.3

===== HASIL =====
Fungsi : f(X) = 1.0X^3 + 1.1339286X^2 + 0.0X^1 + 0.0
Taksiran f(1.3) = 2.77410718

>>> Apakah ingin disimpan dalam file? [y/n]:
```

Contoh input salah:

```
===== MENU UTAMA =====
1. Sistem Persamaan Linier
2. Determinan
3. Matriks balikan
4. Interpolasi Polinom
5. Interpolasi Bicubic Spline
6. Regresi linier berganda
7. Pembesaran Gambar dengan Interpolasi Bicubic Spline
8. Keluar
>>> Pilih menu : 4

===== Pilih Metode Masukkan =====
1. Masukkan Ketik
2. Masukkan dalam bentuk File
3. Kembali
>>> Pilih Metode Masukkan : 1

===== Metode Ketik =====
Ketik -1 Kembali
>>> Masukkan Jumlah Titik : 4
Masukkan Titik - Titik (format: x y)
a
Masukkan Titik salah.
0.5 0.5
0.5 0.5
Titik dengan absis 0.5 sudah pernah diberikan. Masukkan titik lain.
```

## 4.5 Regresi Linier Berganda

Diberikan sekumpulan data sesuai pada tabel berikut ini.

Table 12.1: Data for Example 12.1

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

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

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

```

File Edit Selection View Go Run Terminal Help
Tubes_Algo_1

test > regress > input > regress.txt
72.4 76.3 29.18 0.90

***** APLIKASI MULAI *****
***** Selamat Datang pada Aplikasi Perhitungan Matriks dan SPL *****

MENU UTAMA
1. Sistem Persamaan Linier
2. Determinan
3. Matriks balikan
4. Interpolasi Polinom
5. Interpolasi Bicubic Spline
6. Regresi linier berganda
7. Pembaruan Gambar dengan Interpolasi Bicubic Spline
8. Keluar
>>> Pilih menu : 6

----- Pilih Metode Masukan -----
1. Masukkan Ketik
2. Masukkan dalam bentuk File
3. Kembali
>>> Pilih Metode Masukan : 2
Ketik 0 untuk kembali
>>> Masukkan nama file: regress.txt

HASIL
72.4 76.3 29.18 0.9
41.6 70.3 29.35 0.91
34.3 77.1 29.24 0.96
35.1 68.0 29.27 0.89
10.7 79.0 29.78 1.0
12.9 67.4 29.39 1.1
8.3 66.8 29.69 1.15
20.1 76.9 29.48 1.03
72.2 77.7 29.09 0.77
24.0 67.7 29.6 1.07
23.2 76.8 29.38 1.07
47.4 86.6 29.35 0.94
31.5 76.9 29.63 1.1
10.6 86.3 29.56 1.1
11.2 86.0 29.48 1.1
73.3 76.3 29.4 0.91
75.4 77.9 29.28 0.87
96.6 78.7 29.29 0.78
107.4 86.8 29.03 0.82
54.9 70.9 29.37 0.95

f(x) = 19.42 + 779.476999999999x1 + 1483.43699999999x2 + 571.121500000000x3
Taksiran = 168468.35366999999
  
```

## 4.6 Bicubic Spline Interpolation

Proses *bicubic spline interpolation* di antara 4 titik *normalize* dengan 12 titik di sekitarnya untuk menentukan turunan berarah setiap titik menjadi perhitungan yang dapat digunakan untuk mengaproksimasi nilai di dalam area *normalize*.

Pada eksperimen ini akan dilakukan perhitungan *bicubic spline interpolation* untuk menentukan  $f(x,y)$  dengan  $0 \leq x,y \leq 1$  (di dalam area *normalize*). Berikut adalah data 16 titik yang akan dihitung.

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

```
>>> Pilih menu : 5

===== Masukkan Bicubic Spline =====
Ketik 0 untuk kembali
>>> Masukkan nama file: testBicubic.txt
===== HASIL =====
a00 = 21.0
a10 = 51.0
a20 = 28.0
a30 = -2.0
a01 = 0.0
a11 = 16.0
a21 = 82.0
a31 = -56.0
a02 = 240.0
a12 = 217.0
a22 = -1295.0
a32 = 709.0
a03 = -136.0
a13 = -123.0
a23 = 888.0
a33 = -487.0

f(0.25,0.75) = 117.732177734375

>>> Apakah ingin disimpan dalam file? [y/n]: y
>>> Masukkan nama file hasil: testBicubic.txt

Penyelesaian berhasil disimpan ke : ../test/bicubic/output/testBicubic.txt
```



```

>>> Pilih menu : 5

===== Masukkan Bicubic Spline =====
Ketik 0 untuk kembali
>>> Masukkan nama file: testBicubic.txt
===== HASIL =====
a00 = 21.0
a10 = 51.0
a20 = 28.0
a30 = -2.0
a01 = 0.0
a11 = 16.0
a21 = 82.0
a31 = -56.0
a02 = 240.0
a12 = 217.0
a22 = -1295.0
a32 = 709.0
a03 = -136.0
a13 = -123.0
a23 = 888.0
a33 = -487.0

f(0.1,0.9) = 128.57518700000003

>>> Apakah ingin disimpan dalam file? [y/n]: y
>>> Masukkan nama file hasil: testBicubic.txt

Penyelesaian berhasil disimpan ke : ../test/bicubic/output/testBicubic(1).txt

```

Setelah melakukan perhitungan interpolasi akan diperoleh nilai  $a_{ij}$  yang kemudian akan digunakan untuk menghitung  $f(x,y)$ . Dari persoalan di atas diperoleh nilai  $f(0.25, 0.75) = 117.732$  dan  $f(0.1, 0.9) = 128.575$ .

#### 4.7 Aplikasi *Bicubic Spline Interpolation* - Peningkatan Kualitas Gambar (ImageBSI)

Pada eksperimen ini sebuah gambar dengan nama `rgbTest3.png` akan dibesarkan dengan skala 300. Dimensi awal `rgbTest.png` adalah 5x5 (lebar 5 px dan tinggi 5px). Dengan perbesaran 300 kali hasil akhir adalah 1500x1500. Berikut ini adalah proses perbesaran gambar.

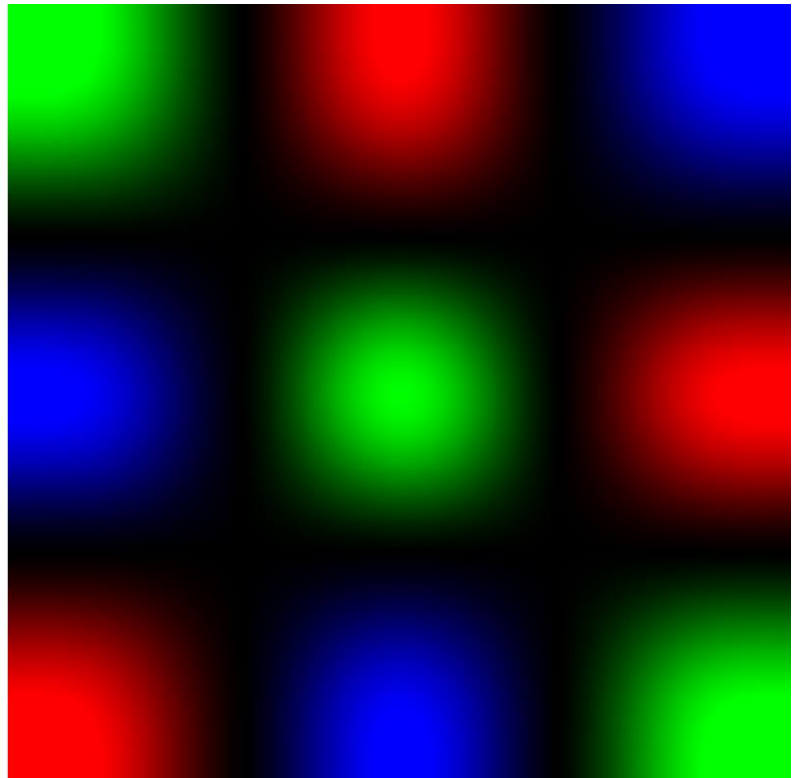
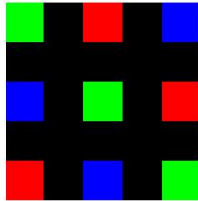
```

>>> Pilih menu : 7

===== MENU Pembesaran Gambar =====
Ketik 0 untuk kembali
>>> Masukkan Nama file sumber: rgbTest3.png
>>> Masukkan nama file output: rgb3Scalled.png
>>> Masukkan skala pembesaran gambar: 300
===== HASIL =====
Gambar sedang diproses ...
( Hasil mungkin membutuhkan waktu lama untuk dicetak walau menu baru sudah keluar )
Gambar Berhasil dibesarkan
Gambar hasil : ../test/imgBSI/output/rgb3Scalled.png

```

Gambar awal (kiri) kemudian akan diperbesar dengan memanfaatkan *Bicubic Spline Interpolation*. Hasil perbesaran (kanan) adalah gambar dengan ukuran 1500x1500.



## **BAB V**

### **KESIMPULAN**

#### **5.1 Kesimpulan**

Mata Kuliah Aljabar Linier dan Geometri mengajarkan berbagai macam metode penyelesaian sistem persamaan linear seperti OBE , Gauss, Gauss – Jordan, kaidah cramer, matrix balikan, dll. Dengan pengetahuan yang didapat, kami bisa memecahkan berbagai macam masalah . Melalui program ini, kami mengimplementasikan ilmu kami untuk menyelesaikan sistem persamaan linear dengan berbagai metode, mencari persamaan polinom interpolasi, menghitung nilai taksiran interpolasi polinom dan bikubik, menghitung hasil regresi linear berganda, dan melakukan scaling image.

#### **5.2 Saran**

- Diperlukan lebih banyak komunikasi seperti update fungsi fungsi yang sudah selesai dan review program yang dibuatkan anggota kelompok agar tidak 2 kali kerja dan program yang dibuat efisien
- Penjelasan fungsi yang dibuat tidak lengkap sehingga kebingungan buat dipakai dan sering error

#### **5.3 Refleksi**

Tugas besar ini memberikan kami berbagai macam pengalaman dan ilmu baru. Kami mempelajari bahasa pemrograman baru, memperdalam pengetahuan kami mengenai ADT. Kami juga menyadari bahwa komunikasi dan kerja sama dalam tim sangatlah penting agar meminimalisir program yang bentrok dan memaksimalkan program dengan fungsi fungsi yang sudah dibuat satu sama lain. Dalam pengerjaan tugas ini, kami juga mengetahui kelemahan dan kelebihan satu sama lain sehingga kami bisa saling melengkapi dan mengambil inisiatif lebih untuk membantu.

## REFERENSI

Mssc.mu.edu (15 Februari 2018). BiLinear, Bicubic, and In Between Spline Interpolation. Diakses pada 30 September, dari [https://www.mssc.mu.edu/~daniel/pubs/RoweTalkMSCS\\_BiCubic.pdf](https://www.mssc.mu.edu/~daniel/pubs/RoweTalkMSCS_BiCubic.pdf)

M. Nazir. 1983. Metode Statistika Dasar 1. Gramedia Pustaka Utama: Jakarta.

Yuliara, I Made. 2016. Regresi Linier Berganda. Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Udayana.