

**Anggota Kelompok:**

Fityatul Haq Rosyidi (13523116)

Reza Ahmad Syarif (13523119)

**Program Studi Teknik Informatika**  
**Institut Teknologi Bandung**

## DAFTAR ISI

1	DESKRIPSI MASALAH.....	6
1.1	Sistem Persamaan Linier (SPL).....	6
1.2	Interpolasi Polinomial.....	6
1.3	Regresi Berganda .....	8
1.3.1	Regresi Linier Berganda .....	8
1.3.2	Regresi Kuadratik Berganda.....	8
1.4	Bicubic Spline Interpolation .....	9
2	TEORI SINGKAT.....	12
2.1	Sistem Persamaan Linier .....	12
2.1.1	Metode Eliminasi Gauss .....	12
2.1.2	Metode Eliminasi Gauss-Jordan.....	12
2.1.3	Metode Matriks Balikan.....	12
2.1.4	Kaidah Cramer .....	13
2.2	Determinan Matriks .....	13
2.2.1	Metode Reduksi Baris .....	13
2.2.2	Metode Ekspansi Kofaktor.....	13
2.3	Balikan Matriks.....	13
2.3.1	Metode Gauss-Jordan.....	13
2.3.2	Metode Adjoin .....	14
2.4	Interpolasi .....	14
2.4.1	Interpolasi Polinom .....	14
2.4.2	Interpolasi Bicubic Spline .....	15
2.5	Regresi.....	15
2.5.1	Regresi Linier Berganda .....	15
2.5.2	Regresi Kuadratik Berganda.....	16
3	IMPLEMENTASI PUSTAKA dan PROGRAM dalam JAVA.....	18
3.1	Implementasi Pustaka .....	18
3.1.1	Matrix.java .....	18
3.1.2	MatrixSolver.java .....	19
3.1.3	InputUtils.java .....	21

3.1.4	OutputUtils.java .....	21
3.1.5	BicubicSplineInterpolation.java .....	22
3.1.6	PolynomialInterpolation.java .....	23
3.1.7	MultipleLinearRegression.java .....	23
3.1.8	MultipleQuadraticRegression.java .....	23
3.1.9	Main.java .....	24
3.2	Struktur Class .....	25
4	EKSPERIMEN .....	26
4.1	Sistem Persamaan Linier .....	26
4.1.1	SPL berbentuk matriks normal .....	26
4.1.2	SPL berbentuk matriks augmented .....	29
4.1.3	SPL dengan kasus.....	31
4.2	Determinan Matriks .....	34
4.2.1	Metode Reduksi Baris .....	34
4.2.2	Metode Ekspansi Kofaktor.....	36
4.3	Balikan Matriks.....	38
4.3.1	Metode Gauss-Jordan.....	38
4.3.2	Metode Adjoin .....	40
4.4	Interpolasi .....	41
4.4.1	Interpolasi Polinom .....	42
4.4.2	Interpolasi Bicubic Spline .....	45
4.4.2.1	.....	45
4.4.2.2	.....	47
4.5	Regresi Linier Berganda .....	48
4.6	Regresi Kuadratik Berganda .....	49
4.7	Image Resizing .....	50
5	KESIMPULAN.....	50
5.1	Kesimpulan.....	50
5.2	Saran .....	50
5.3	Komentar .....	51
5.4	Refleksi.....	51
6	LAMPIRAN.....	52
6.1	Referensi .....	52

6.2	Tautan Repository Github .....	52
6.3	Tautan Video .....	52

## DAFTAR GAMBAR

Gambar 1. Eliminasi Gauss dilakukan dengan matriks eselon baris dan eliminasi Gauss-Jordan dengan matriks eselon baris tereduksi. ....	6
Gambar 2. Ilustrasi beberapa titik yang diinterpolasi secara polinomial. ....	7
Gambar 3. Pemodelan interpolasi <i>bicubic spline</i> . ....	9
Gambar 4. Nilai fungsi yang akan di interpolasi pada titik merah, turunan berarah terhadap sumbu $x$ , terhadap sumbu $y$ , dan keduanya (kiri ke kanan). ....	11
Gambar 2.1.1 Sistem persamaan linier yang akan diubah menjadi augmented matrix. ....	12
Gambar 2.1.2 Perbedaan bentuk eselon biasa dengan bentuk eselon baris tereduksi. ....	12
Gambar 4.4.2.1 Isi dari file studi_kasus.txt .....	42

## 1 DESKRIPSI MASALAH

### 1.1 Sistem Persamaan Linier (SPL)

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.

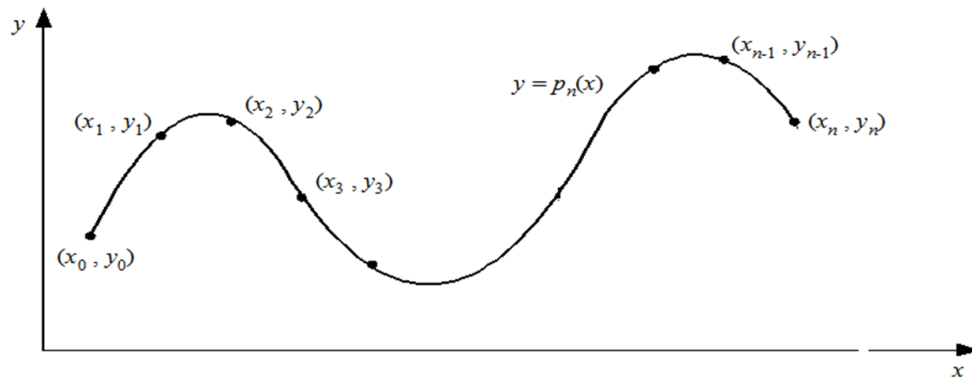
Di dalam Tugas Besar 1 ini, Anda diminta membuat satu atau lebih *library* aljabar linier dalam Bahasa Java. Library tersebut berisi fungsi-fungsi seperti eliminasi Gauss, eliminasi Gauss-Jordan, menentukan balikan matriks, menghitung determinan, kaidah *Cramer* (kaidah *Cramer* khusus untuk SPL dengan  $n$  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-libraryapi>

### 1.2 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  $a_0, a_1, a_2, \dots, a_n$ ,

$$a_0 + a_1x_0 + a_2x_0^2 + \dots + a_nx_0^n = y_0$$

$$a_0 + a_1x_1 + a_2x_1^2 + \dots + a_nx_1^n = y_1$$

...

$$a_0 + a_1x_n + a_2x_n^2 + \dots + a_nx_n^n = y_n$$

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 kuadrat lalu estimasi nilai fungsi pada  $x = 9.2$ . Polinom kuadrat 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

$$a_0 + 8.0a_1 + 64.00a_2 = 2.0794$$

$$a_0 + 9.0a_1 + 81.00a_2 = 2.1972$$

$$a_0 + 9.5a_1 + 90.25a_2 = 2.2513$$

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

### 1.3 Regresi Berganda

Regresi (akan dipelajari lebih lanjut di Probabilitas dan Statistika) merupakan salah satu metode untuk memprediksi nilai selain menggunakan Interpolasi Polinom. Pada tugas besar ini, anda diminta untuk membuat 2 jenis regresi yaitu Regresi Linier Berganda dan Regresi Kuadratik Berganda.

#### 1.3.1 Regresi Linier Berganda

Meskipun sudah ada persamaan jadi untuk menghitung regresi linear sederhana, terdapat persamaan umum dari regresi linear yang bisa digunakan untuk regresi linear berganda, yaitu.

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

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

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

#### 1.3.2 Regresi Kuadratik Berganda

Dalam kasus ini, proses mengubah data-data dalam regresi kuadratik berganda cukup berbeda dengan Regresi Linier Berganda. Bentuk persamaan dari regresi kuadratik ada 3, yaitu:

- Variabel Linier: Variabel dengan derajat satu seperti X, Y, dan Z
- Variabel Kuadrat: Variabel dengan derajat dua seperti X<sup>2</sup>
- Variabel Interaksi: 2 Variabel dengan derajat satu yang dikalikan dengan satu sama lain seperti XY, YZ, dan XZ

Setiap n-peubah, jumlah variabel linier, kuadrat, dan interaksi akan berbeda-beda. Perhatikan contoh regresi kuadratik 2 variabel peubah sebagai berikut!



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

N menandakan jumlah peubah, terdapat 2 variabel linier yaitu  $u_i$  dan  $v_i$ , 2 variabel kuadrat yaitu  $u_i^2$  dan  $v_i^2$ , dan 1 variabel interaksi yaitu  $u_i v_i$ . Untuk setiap  $n$ -peubah, akan terdapat 1 konstan  $N$  (Terlihat di bagian atas kiri gambar),  $n$  variabel linier,  $n$  variabel kuadrat, dan  $C_2 n$  variabel linier (dengan syarat  $n > 1$ ). Tentu dengan bertambahnya peubah  $n$ , ukuran matriks akan bertambah lebih besar dibandingkan regresi linier berganda tetapi solusi tetap bisa didapat dengan menggunakan SPL.

Kedua model regresi yang dijadikan sistem persamaan linier tersebut diselesaikan dengan menggunakan metode eliminasi Gauss.

#### 1.4 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.

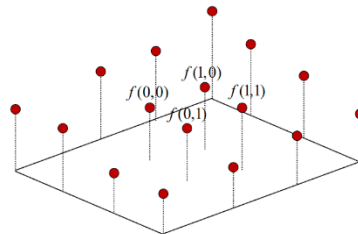
Dalam pemrosesan menggunakan interpolasi *bicubic spline* digunakan 16 buah titik, 4 titik referensi utama di bagian pusat, dan 12 titik di sekitarnya sebagai aproksimasi turunan dari keempat titik referensi untuk membangun permukaan bikubik. Bentuk pemodelannya adalah sebagai berikut.

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.

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

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

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

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

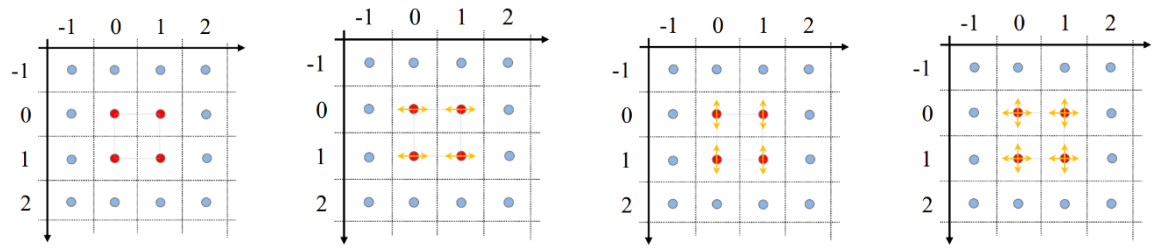
Dengan menggunakan nilai fungsi dan turunan berarah tersebut, dapat terbentuk sebuah matriks solusi  $X$  yang membentuk persamaan penyelesaian sebagai berikut.

$$y = Xa$$

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

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  $f_x(1, 1)$  sehingga diperoleh nilai konstanta  $1 \times 1^{1-1} \times 1^0 = 1$ , sesuai dengan isi matriks  $X$ .

Nilai dari vektor  $a$  dapat dicari dari persamaan  $y = Xa$ , lalu vektor  $a$  tersebut digunakan sebagai nilai variabel dalam  $f(x, y)$ , sehingga terbentuk fungsi interpolasi bicubic sesuai model. Tugas Anda pada studi kasus ini adalah membangun persamaan  $f(x, y)$  yang akan digunakan untuk melakukan interpolasi berdasarkan nilai  $f(a, b)$  dari masukan matriks  $4 \times 4$ . 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 di interpolasi pada titik merah, turunan berarah terhadap sumbu  $x$ , terhadap sumbu  $y$ , dan keduanya (kiri ke kanan).

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

## 2 TEORI SINGKAT

### 2.1 Sistem Persamaan Linier

Sebuah paragraf template.

#### 2.1.1 Metode Eliminasi Gauss

Metode eliminasi Gauss adalah sebuah metode untuk memecahkan sistem persamaan linier yang berbentuk matriks teraugmentasi (*augmented matrix*).

$$\begin{array}{rcl} a_{11}x_1 + a_{12}x_2 + a_{13}x_3 + \dots + a_{1n}x_n = b_1 & \dots & (1) \\ a_{21}x_1 + a_{22}x_2 + a_{23}x_3 + \dots + a_{2n}x_n = b_2 & \dots & (2) \\ a_{31}x_1 + a_{32}x_2 + a_{33}x_3 + \dots + a_{3n}x_n = b_3 & \dots & (3) \\ \vdots & \vdots & \vdots \\ a_{n1}x_1 + a_{n2}x_2 + a_{n3}x_3 + \dots + a_{nn}x_n = b_n & \dots & (n) \end{array}$$

Gambar 2.1.1

Sistem persamaan linier yang akan diubah menjadi augmented matrix.

Matriks yang teraugmentasi diubah menjadi sebuah matriks segitiga atas (bentuk eselon) lalu dicari solusinya menggunakan metode penyulihan balik.

#### 2.1.2 Metode Eliminasi Gauss-Jordan

Metode eliminasi Gauss-Jordan melanjutkan metode eliminasi Gauss dengan mengubah semua elemen yang ada di atas tiap satu utama menjadi 0 juga. Melalui metode ini, diperoleh bentuk *reduced row echelon form* atau *bentuk eselon baris tereduksi*. Melalui metode ini, tidak perlu dilakukan penyulihan balik.

Echelon form	Reduced echelon form
$\begin{bmatrix} a & b & c \\ 0 & d & e \\ 0 & 0 & f \end{bmatrix}$	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

Gambar 2.1.2

Perbedaan bentuk eselon biasa dengan bentuk eselon baris tereduksi.

#### 2.1.3 Metode Matriks Balikan

Metode matriks balikan dapat digunakan untuk memecahkan sebuah sistem persamaan linier dengan mencari invers dari matriks koefisien. Idanya berdasarkan pada fakta bahwa jika A adalah matriks koefisien dari sebuah sistem persamaan linier, dan A punya inverse, maka solusi dari sistem  $Ax = b$  dapat ditemukan dengan mengalikan invers A dengan b (vektor konstanta).

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

#### 2.1.4 Kaidah Cramer

Kaidah Cramer adalah sebuah cara untuk memecahkan sebuah sistem persamaan linier dengan menggunakan determinan matriks. Langkah-langkah yang dilakukan pada kaidah cramer:

1. Menuliskan sistem persamaan linier dalam bentuk matriks  $Ax = b$ .
2. Menghitung determinan dari matriks koefisien A.
3. Untuk tiap  $x_i$  yang tidak diketahui, menggantikan kolom ke-i dari matriks A dengan vektor konstanta untuk membentuk sebuah matriks baru  $A_i$ .
4. Menghitung determinan dari matriks baru tersebut.
5. Menghitung nilai dari x yang tak diketahui dengan rumus:

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

### 2.2 Determinan Matriks

#### 2.2.1 Metode Reduksi Baris

Metode reduksi baris adalah sebuah metode yang menggunakan eliminasi Gauss untuk mengubah matriks menjadi bentuk segitiga atas. Setelah itu, semua elemen diagonal dari matriks dikalikan untuk menemukan determinan dari matriks tersebut.

#### 2.2.2 Metode Ekspansi Kofaktor

Metode ekspansi kofaktor adalah sebuah metode yang menghitung determinan suatu matriks melalui metode rekursif dengan menghitung determinan-determinan dari submatriks. Langkah-langkah yang dilakukan yaitu:

1. Memilih sebuah baris atau kolom
2. Mengaplikasikan rumus kofaktor

$$\det(A) = \sum_{j=1}^n a_{ij} * C_{ij}$$

di mana  $A_{ij}$  adalah elemen-elemen dari baris yang dipilih dan  $C_{ij}$  adalah kofaktornya. Rumus dari kofaktor adalah

$$C_{ij} = (-1)^{i+j} * \det(A_{ij})$$

di mana  $A_{ij}$  adalah submatriks yang diperoleh dengan menghapus baris ke-I dan kolom ke-j.

### 2.3 Balikan Matriks

#### 2.3.1 Metode Gauss-Jordan

Metode Gauss-Jordan dilakukan dengan melakukan operasi-operasi baris elementer pada sebuah matriks untuk mengubahnya menjadi inversnya. Cara ini

dilakukan dengan mengaugmentasikan matriks A yang ingin diinverskan dengan sebuah matriks identitas yang berukuran sama.

$$\left[ A \mid I \right]$$

Gambar 2.3.1.1

Matriks yang teraugmentasi sebelum dilakukan rangkaian OBE.

Lalu, dilakukan operasi-operasi baris pada matriks teraugmentasi tersebut sampai sisi kiri berubah menjadi matriks identitas. Jika berhasil, maka matriks identitas di sisi kanan akan berubah menjadi matriks invers dari A.

$$\left[ I \mid A^{-1} \right]$$

Gambar 2.3.1.2

Matriks setelah dilakukan rangkaian OBE.

### 2.3.2 Metode Adjoin

Metode adjoin adalah sebuah metode untuk mencari invers dari sebuah matriks dengan memanfaatkan konsep adjoin. Langkah-langkah yang dilakukan.

1. Memastikan determinan dari matriks A bukan 0 karena invers hanya ada jika  $\det(A)$  tidak sama dengan 0.
2. Mencari adjoin dari matriks A. Adjoin adalah matriks A adalah transpose dari matriks kofaktor. Untuk menghitung kofaktor dari tiap elemen  $a_{ij}$  di matriks A dapat dilakukan dengan
  - a. Menghapus baris ke-i dan kolom ke-j untuk membentuk submatriks  $A_{ij}$
  - b. Menghitung determinan dari matriks  $A_{ij}$  dan mengalikannya dengan  $(-1)^{i+j}$  untuk mencari kofaktor dari  $C_{ij}$ .

Kofaktor-kofaktor ini dapat disusun menjadi sebuah matriks lalu diambil transpose-nya untuk memperoleh  $\text{adj}(A)$ .

3. Inverse dari matriks A dapat diperoleh dengan rumus:

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

## 2.4 Interpolasi

### 2.4.1 Interpolasi Polinom

Interpolasi polinomial adalah sebuah proses untuk mencari polinomial  $P(x)$  dengan derajat  $n - 1$  yang akan melewati  $n$  titik data  $(x_0, y_0), (x_1, y_1), \dots, (x_{n-1}, y_{n-1})$  secara halus. Polinomial dipresentasikan dalam bentuk:

$$P(x) = a_0 + a_1x + a_2x^2 + \dots + a_{n-1}x^{n-1}$$

Dengan cara tersebut, dapat dibuat 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  $a_0, a_1, a_2, \dots, a_n$ ,

$$a_0 + a_1x_0 + a_2x_0^2 + \dots + a_nx_0^n = y_0$$

$$a_0 + a_1x_1 + a_2x_1^2 + \dots + a_nx_1^n = y_1$$

...

$$a_0 + a_1x_n + a_2x_n^2 + \dots + a_nx_n^n = y_n$$

Persamaan ini bisa dipecahkan menggunakan metode eliminasi Gauss atau Gauss-Jordan.

#### 2.4.2 Interpolasi Bicubic Spline

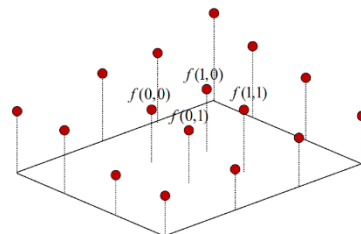
*Bicubic spline interpolation* adalah metode interpolasi yang digunakan untuk mengaproksimasi fungsi di antara titik-titik data yang diketahui. *Bicubic spline interpolation* melibatkan konsep *spline* dan konstruksi serangkaian polinomial kubik di dalam setiap sel segi empat dari data yang diberikan.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

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:  $a_{ij}$



## 2.5 Regresi

### 2.5.1 Regresi Linier Berganda

Regresi linier berganda adalah sebuah ekstensi dari regresi linier sederhana yang memodelkan hubungan antara dua atau lebih variabel independen dan sebuah variabel dependen. Hubungan antara variabel dependen  $y$  dengan variabel-variabel independen dimodelkan berdasarkan persamaan berikut:

$$y = \beta_0 + \beta_1x_{1i} + \beta_2x_{2i} + \dots + \beta_kx_{ki} + \epsilon_i$$

Untuk mendapatkan nilai dari setiap  $\beta_i$  dapat digunakan *Normal Estimation Equation for Multiple Regression*.

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

### 2.5.2 Regresi Kuadratik Berganda

Regresi kuadratik berganda adalah teknik analisis statistik yang digunakan untuk menilai hubungan antara satu variabel dependen dan dua atau lebih variabel independen. Model ini menggabungkan variabel linier, kuadrat, dan interaksi, yang memungkinkan analisis hubungan yang lebih kompleks. Model tersebut dapat ditulis sebagai

$$y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_1^2 + \beta_4 X_2^2 + \beta_5 X_1 X_2$$

#### 1. Variabel Linear

Variabel ini memiliki derajat satu, seperti  $X_1$  dan  $X_2$ . Variabel ini berkontribusi langsung terhadap variabel dependen tanpa adanya pembangkitan kuadrat atau interaksi. Dalam model regresi, variabel linier memiliki bentuk dasar yang berkontribusi dengan koefisien tertentu.

#### 2. Variabel Kuadrat

Variabel dengan derajat dua, seperti  $X_1^2$  dan  $X_2^2$ . Variabel kuadrat digunakan untuk menangkap efek non-linear dari variabel independen terhadap variabel dependen. Penambahan variabel kuadrat dapat membantu dalam mendeteksi pola atau kurva yang lebih kompleks dalam data.

#### 3. Variabel Interaksi

Variabel ini dihasilkan dari perkalian dua variabel linier, seperti  $X_1 X_2$ . Variabel interaksi memungkinkan model untuk menangkap efek sinergis antara dua variabel, di mana pengaruh satu variabel terhadap hasil tergantung pada nilai variabel lainnya.

Dalam model kuadratik berganda, estimasi koefisien regresi dilakukan menggunakan metode Ordinary Least Squares (OLS). Metode ini bertujuan untuk menemukan nilai koefisien  $\beta$  yang meminimalkan jumlah kuadrat dari residu, yaitu selisih antara nilai yang diprediksi dan nilai observasi sebenarnya.

Prosedur OLS dapat dirangkum sebagai berikut:



1. **Matriks Desain:** Matriks desain  $X$  disusun, yang berisi semua variabel independen (termasuk variabel kuadrat dan interaksi) serta kolom untuk intercept.
2. **Estimasi Koefisien:** Estimasi koefisien regresi diberikan oleh rumus:

$$\beta = (X^T X)^{-1} X^T y$$

$y$  adalah vektor hasil observasi.

Dengan menggabungkan variabel linier, kuadrat, interaksi, dan menerapkan metode OLS, regresi kuadratik berganda mampu menangkap hubungan yang kompleks dan memberikan gambaran yang lebih akurat tentang interaksi faktor-faktor tersebut.

### 3 IMPLEMENTASI PUSTAKA dan PROGRAM dalam JAVA

#### 3.1 Implementasi Pustaka

Pustaka yang dibuat berisi Tipe Data Abstrak (ADT) Matriks dan fungsi-fungsi primitifnya. ADT Matriks akan digunakan diberbagai fungsi-fungsi lainnya. Fungsi-fungsi lain seperti menghitung determinan, menentukan balikan matriks, solusi persamaan linear menggunakan eliminasi Gauss, eliminasi Gauss-Jordan, balikan invers, dan kaidah Cramer (khusus untuk SPL dengan n peubah dan n persamaan). Digunakan juga pada interpolasi dan regresi.

##### 3.1.1 Matrix.java

```
// ATRIBUT MATRIKS
public double[][] data; // elemen matriks
public int rowEff; // jumlah baris efektif
public int colEff; // jumlah kolom efektif

// Konstruktor matriks
public Matrix(int nRows, int nCols)

// Primitif MATRIKS
// Mengatur elemen matriks pada indeks (row, col) dengan nilai
value
void setElement(int row, int col, double value);

// Mengecek apakah indeks (i, j) valid dalam batas matriks (asumsi
maksimal 100x100)
boolean isMatrixIdxValid(int i, int j);

// Mendapatkan indeks terakhir dari baris efektif matriks
int getLastIdxRow();

// Mendapatkan indeks terakhir dari kolom efektif matriks
int getLastIdxCol();

// Mengecek apakah indeks efektif (i, j) valid untuk matriks saat
ini
boolean isIdxEff(int i, int j);

// Mengambil elemen diagonal dari matriks (khusus matriks persegi)
double getElmtDiagonal(int i);

// Menyalin isi matriks dari matriks lain
void copyMatrix(Matrix mIn);

// Membaca matriks dari input pengguna
Matrix readMatrixFromInput(Scanner sc);

// Membaca matriks dari file eksternal
Matrix readMatrixFromFile(String filename) throws IOException;

// Menampilkan matriks ke layar dengan pembulatan 4 angka di
belakang koma
void displayMatrix();
```

```

// Menjumlahkan dua matriks (ukuran harus sama)
Matrix addMatrix(Matrix m2);

// Mengurangkan dua matriks (ukuran harus sama)
Matrix subtractMatrix(Matrix m2);

// Mengalikan dua matriks (baris pertama sama dengan kolom kedua)
Matrix multiplyMatrix(Matrix m2);

// Mengalikan matriks dengan skalar
Matrix multiplyByConst(double x);

// Mengecek apakah dua matriks sama
boolean isMatrixEqual(Matrix m2);

// Mengecek apakah matriks adalah matriks persegi
boolean isSquare();

// Mengecek apakah matriks simetris
boolean isSymmetric();

// Mengecek apakah matriks identitas (diagonal 1, lainnya 0)
boolean isIdentity();

// Mengecek apakah matriks jarang (sparse)
boolean isSparse();

// Mentransposisi matriks
Matrix transpose();

// Mengisi nilai matriks dengan array 2D
void fillMatrix(double[][] values);

// OBE: Menukar dua baris dalam matriks
void swapRows(int row1, int row2);

// OBE: Mengalikan baris dengan konstanta
void multiplyRowByConstant(int row, double constant);

// OBE: Menambahkan kelipatan baris lain ke baris target
void addMultipleOfRow(int targetRow, int sourceRow, double
multiple);

// Memisahkan matriks augmented menjadi dua bagian X dan Y
void splitAugmentedMatrix(double[][] augmentedMatrix, double[][]
X, double[][] Y);

```

### 3.1.2 MatrixSolver.java

```

// Toleransi untuk angka yang sangat kecil
private static final double epsilon = 1e-10;

// Determinan matriks dengan metode ekspansi kofaktor
public static double determinant(Matrix matrix)

// Fungsi menghitung determinan dengan metode ekspansi kofaktor

```

```

private static double determinantByCofactorExpansion(double[][]
matrix)

// Fungsi untuk menghitung cofactor
private static double[][] minor(double[][] matrix, int
excludingRow, int excludingCol)

// Determinan matriks dengan metode reduksi baris
public static double determinantByRowReduction(Matrix matrix)

// Fungsi untuk menghitung adjoin matriks
public static Matrix adjoin(Matrix matrix)

// Fungsi untuk menghitung balikan matriks
public static Matrix inverseAdjoin(Matrix matrix)

// Fungsi untuk menghitung invers matriks menggunakan metode
Gauss-Jordan
public static Matrix inverseGaussJordan(Matrix matrix)

// Fungsi untuk menyelesaikan SPL dengan matriks balikan
menggunakan matriks augmented
public static double[] solveUsingInverse(Matrix matrix)

// Fungsi untuk mengisi matriks
public static void fillMatrix(Matrix matrix, Matrix values)

// Fungsi untuk mengalikan matriks (Matrix) dengan vektor
(double[])
public static double[] multiplyMatrixWithVector(Matrix matrix,
double[] vector)

// Fungsi untuk menyelesaikan SPL dengan kaidah Cremer menggunakan
matriks augmented
public static double[] solveUsingCramer(Matrix matrix)

// Fungsi untuk mengganti kolom pada matriks A dengan vektor B
private static Matrix substituteColumn(Matrix A, double[] B, int
columnIndex)

// Fungsi untuk memotong elemen terakhir baris
private static double[] truncateLastCol(double[] row)

// Fungsi untuk mengecek apakah elemen baris adalah nol semua
private static boolean isRowZero(double[] row)

// Fungsi Eliminasi gauss jordan
public static double[] gaussJordanElimination(Matrix matrix)

// Fungsi Eliminasi gauss jordan
public static String gaussJordanEliminationForMain(Matrix matrix)

// Fungsi Eliminasi gauss jordan
public static String gaussEliminationForMain(Matrix matrix) }

```

### 3.1.3 InputUtils.java

```
private static Scanner scanner = new Scanner(System.in);

// BASIC UTILITIES
// Metode untuk meminta integer dari pengguna
public static int getInt(String prompt)

// Metode untuk meminta double dari pengguna
public static double getDouble(String prompt)

// Metode untuk meminta string dari pengguna
public static String getString(String prompt)

// COMPLEX UTILITIES
public static Matrix readMatrixFromInput()

// Membaca matriks dari file
public static Matrix readMatrixFromFile(String folder, String
filename)

public static Matrix readMatrixFromFileForBicubic(String filepath)

public static Object[] readVectorAndABfromFile(String filename)

// Metode untuk meminta matriks dari pengguna
public static double[][] getMatrix(int rows, int cols, String
prompt)

public static double[][] getXYdata(int n, String prompt)

// Metode untuk meminta array 1 dimensi (untuk konstanta, atau
untuk nilai x dan y.)
public static double[] getArray(int size, String prompt)

public static double[][] readAugmentedMatrixFromKeyboard(int m,
int n)

public static double[][] readAugmentedMatrixFromFile(String
filePath, int n)

public static Matrix readMatrixFromKeyboard()

public static double[][] readMatrixFromFile2(String filePath)

public static double[][] readAugmentedMatrix(double[][]
augmentedMatrix)

public static void pressAnyKeyToProceed()
```

### 3.1.4 OutputUtils.java

```
// BASIC UTILITIES
```

```

// Menampilkan matriks ke layar dengan pembulatan empat angka
belakang koma
public static void displayMatrix(Matrix matrix)

// Metode untuk mencetak sebuah array
public static void printArray(double[] array, String arrayName)

// Metode untuk mencetak solusi persamaan linier
public static void printCoefficients(double[] solution)

// Metode untuk mengetak persamaan regresi
public static void printRegressionEquation(double[] regression)

// OUTPUT KE FILE UNTUK TIAP FUNGSI
public static void saveMatrixToFile(Matrix matrix, String
filePath)

public static void saveSistemPersamaanLinier(double[] solution,
String filePath)

public static void saveSistemPersamaanLinierGauss(String solution,
String filePath)

public static void saveDeterminant(double determinant, String
filePath)

public static void saveInverseMatrix(Matrix matrix, String
filePath)

public static void saveInterpolasiPolinom(String equation, String
estimation_output)

public static void saveBicubicSpline(double res)

public static void saveRegresiLinier(String equation, String
estimationOutput)

public static void saveRegresiKuadratikBerganda(String equation,
double[][] observation, double prediction)

```

### 3.1.5 BicubicSplineInterpolation.java

```

// kalkulasi bicubic interpolation
public double calculate(Matrix vectorY, double a, double b,
boolean imageResizing)

// Method untuk membaca gambar dan mengubahnya ke RGB array 3D
public static int[][][] loadImage(String filePath)

// Method untuk menyimpan gambar dari RGB array
public static void saveImage(int[][][] rgbArray, String
outputPath)

```

```
// Method untuk interpolasi bicubic pada blok 4x4
public double calculateBicubic(double[][] block, double x, double
y) {

// Method untuk melakukan scaling pada gambar
public int[][][] scaleImage(int[][][] image, int newWidth, int
newHeight) {
```

### 3.1.6 PolynomialInterpolation.java

```
public class PolynomialInterpolation

public static double[] calculatePolynomialEquation(int n,
double[][] tuples)

public static double calculateY(double[] coefficients, double x)

public static String getPolynomialEquation(double[] coefficients,
int numFeatures)
```

### 3.1.7 MultipleLinearRegression.java

```
// Fungsi untuk menghitung persamaan regresi linear
public static double[] calculateRegressionEquation(double[][]
data_array, int m, int n)

// Fungsi untuk menghitung nilai y berdasarkan persamaan regresi
linear
public static double calculateY(double[] coefficients, double[]
x_array)

// Fungsi untuk mendapatkan persamaan regresi linear sebagai
string
public static String getLinearRegressionEquation(double[]
coefficients, int numFeatures)
```

### 3.1.8 MultipleQuadraticRegression.java

```
// Scanner untuk input pengguna
public static Scanner scanner = new Scanner(System.in);

// Fungsi untuk membangun matriks desain kuadratik
public static Matrix buildQuadraticDesignMatrix(double[][] X)

// Fungsi untuk menghitung koefisien regresi kuadratik
public static Matrix calculateBetaQuadraticRegression(Matrix
Quadratic, double[][] Y)

// Fungsi untuk mencetak persamaan regresi kuadratik
public static void printQuadraticEquation(Matrix coefficients, int
numFeatures)
```

```

// Fungsi untuk memprediksi nilai berdasarkan model regresi
kuadratik
public static double predictQuadraticRegression(Matrix
coefficients, Matrix observation)

// Melatih model regresi kuadratik
public static Matrix trainQuadraticModel(double[][] data_array,
int n)

// Fungsi untuk mendapatkan persamaan regresi kuadratik sebagai
string
public static String getQuadraticEquation(Matrix coefficients, int
numFeatures)

// Fungsi untuk mengambil input observasi baru
public static double[][] inputObservation(int n)

// Fungsi untuk memprediksi nilai berdasarkan model regresi
kuadratik
public static double predictQuadratic(Matrix coefficients,
double[][] observation)

```

### 3.1.9 Main.java

```

// FUNGSI-FUNGSI UNTUK TIAP PILIHAN
// Fungsi untuk Sistem Persamaan Linier
public static void SistemPersamaanLinier()

// Fungsi untuk Determinan
public static void Determinan()

// Fungsi untuk Matriks Balikan
public static void MatriksBalikan()

// Fungsi untuk Interpolasi Polinom
public static void InterpolasiPolinom()

// Fungsi untuk Interpolasi Bicubic Spline
public static void InterpolasiBicubicSpline()

// Fungsi untuk Regresi Linier Berganda
public static void RegresiLinier()

// Fungsi untuk Regresi Kuadratik Berganda
public static void RegresiKuadratikBerganda()

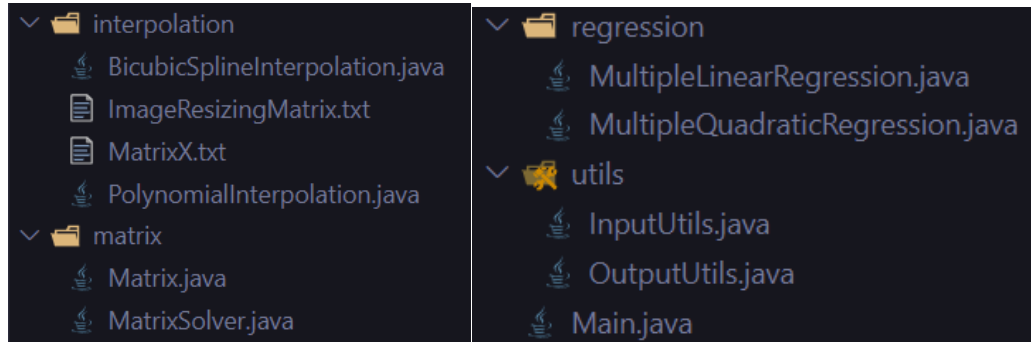
// Fungsi untuk Interpolasi Gambar
public static void ImageResizing()

```



### 3.2 Struktur Class

Struktur dari class yang dibuat dimasukkan dalam folder yang telah disesuaikan dengan tugasnya. Secara garis besar, class terbagi menjadi empat folder yaitu matrix, interpolation, regression, dan utils. Terdapat juga class main sebagai program utama.



Untuk folder matrix terdapat class Matrix yang berisi library berbagai fungsi primitif untuk matriks. Selain itu, terdapat juga class MatrixSolver yang berisi fungsi untuk menyelesaikan persoalan matriks seperti determinan, invers, dan sistem persamaan linier (SPL). Class Matrix dan MatrixSolver ini sering digunakan pada class-class yang lain sebagai alat untuk menemukan solusi permasalahan masing-masing class. Folder interpolation digunakan untuk menyelesaikan permasalahan interpolasi. Di dalam folder interpolation terdapat class BicubicSplineInterpolation dan class PolynomialInterpolation. Folder regression digunakan untuk menyelesaikan permasalahan regresi. Di dalam folder regression terdapat class MultipleLinearRegression dan class MultipleQuadraticRegression.

Folder utils digunakan sebagai utility tambahan dalam proses menemukan solusi setiap permasalahan. Dalam folder ini, terdapat class InputUtils yang berisi berbagai tools untuk melakukan input seperti input matriks augmented dan matriks nxn baik dari keyboard maupun file. Sedangkan, class OutputUtils berisi berbagai tools untuk melakukan output seperti display matrix, print coefficient, dan beberapa fungsi untuk melakukan save ke file. Lalu, untuk class main berfungsi sebagai program utama yang mengintegrasikan berbagai class dan fungsi yang telah dibuat.

## 4 EKSPERIMEN

### 4.1 Sistem Persamaan Linier

Untuk pengujian program dalam penyelesaian SPL, diberikan beberapa studi kasus.

#### 4.1.1 SPL berbentuk matriks normal

##### a. Kasus 1

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

Potongan Akhir penyelesaian program:

```
Masukkan elemen matriks:
1 1 -1 -1 1
2 5 -7 -5 -2
2 -1 1 3 4
5 2 -4 2 6
Metode yang ingin digunakan?
1. Eliminasi Gauss
2. Eliminasi Gauss-Jordan
3. Matriks Balikan
4. Cramer
Masukkan pilihanmu: 1
Tidak ada Solusi
```

Analisis: langkah penyelesaian SPL menggunakan metode gauss dapat menangani persamaan yang tidak konsisten, sehingga dapat menangani apabila SPL tidak memiliki solusi

##### b. Kasus 2

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

Potongan akhir penyelesaian program:

```

Masukkan elemen matriks:
1 -1 0 0 1 3
1 1 0 -3 0 6
2 -1 0 1 -1 5
-1 2 0 -2 -1 -1
Metode yang ingin digunakan?
1. Eliminasi Gauss
2. Eliminasi Gauss-Jordan
3. Matriks Balikan
4. Cramer
Masukkan pilihanmu: 2
X1 = 3.0 + X5
X2 = 0.0 + 2.0X5
X4 = -1.0 + X5
X3 = free
X5 = free

```

Analisis: prosedur penyelesaian SPL menggunakan metode gauss dapat menangani persamaan yang memiliki solusi banyak.

c. Kasus 3

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

Potongan akhir penyelesaian program :

d.

```

Masukkan elemen matriks:
0 1 0 0 1 0 2
0 0 0 1 1 0 -1
0 1 0 0 0 1 1
Metode yang ingin digunakan?
1. Eliminasi Gauss
2. Eliminasi Gauss-Jordan
3. Matriks Balikan
4. Cramer
Masukkan pilihanmu: 2
X2 = 1.0 - X6
X4 = -2.0 - X6
X5 = 1.0 + X6
X1 = free
X3 = free
X6 = free

```

Analisis : program penyelesaian SPL menggunakan Gauss jordan dapat menangani kasus parametrik seperti diatas

e. Kasus 4

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

1. n = 6:

Pada test case ini, masukan berasal dari file bernama SPL4 yang berisi matriks sebagai berikut :

```
1 1.0 0.5 0.3333333333333333 0.25 0.2 0.1666666666666666 1
2 0.5 0.3333333333333333 0.25 0.2 0.1666666666666666 0.14285714285714285 0
3 0.3333333333333333 0.25 0.2 0.1666666666666666 0.14285714285714285 0.125 0
4 0.25 0.2 0.1666666666666666 0.14285714285714285 0.125 0.1111111111111111 0
5 0.2 0.1666666666666666 0.14285714285714285 0.125 0.1111111111111111 0.1 0
6 0.1666666666666666 0.14285714285714285 0.125 0.1111111111111111 0.1 0.09090909090909091 0
```

```
== SISTEM PERSAMAAN LINIER ==
Input dari keyboard atau file?
1. Keyboard
2. File
Masukkan pilihanmu: 2
Masukkan nama file: SPL4
Metode yang ingin digunakan?
1. Eliminasi Gauss
2. Eliminasi Gauss-Jordan
3. Matriks Balikan
4. Cramer
Masukkan pilihanmu: 2
X1 = 36.000000000980656
X2 = -630.0000000292673
X3 = 3360.000000203484
X4 = -7560.000000539232
X5 = 7560.000000603351
X6 = -2772.000000240222
```

Analisis : penyelesaian SPL dengan metode gauss jordan dapat menyelesaikan persoalan matriks hilbert dengan n = 6.

2. n = 10:

Pada test case ini, input berasal dari file bernama SPL5 dengan isi sebagai berikut :

```
1.0 0.5 0.333333333333333 0.25 0.2 0.166666666666666 0.14285714285714285 0.125 0.111111111111111 0.1
0.5 0.333333333333333 0.25 0.2 0.166666666666666 0.14285714285714285 0.125 0.111111111111111 0.1 0.09090909090909091 0.08333333333333333
0.333333333333333 0.25 0.2 0.166666666666666 0.14285714285714285 0.125 0.111111111111111 0.1 0.09090909090909091 0.08333333333333333 0.07692307692307693
0.25 0.2 0.166666666666666 0.14285714285714285 0.125 0.111111111111111 0.1 0.09090909090909091 0.08333333333333333 0.07692307692307693 0.07142857142857142
0.2 0.166666666666666 0.14285714285714285 0.125 0.111111111111111 0.1 0.09090909090909091 0.08333333333333333 0.07692307692307693 0.07142857142857142 0.06666666666666667
0.166666666666666 0.14285714285714285 0.125 0.111111111111111 0.1 0.09090909090909091 0.08333333333333333 0.07692307692307693 0.07142857142857142 0.06666666666666667 0.0625
0.14285714285714285 0.125 0.111111111111111 0.1 0.09090909090909091 0.08333333333333333 0.07692307692307693 0.07142857142857142 0.06666666666666667 0.0625 0.058823529411764705
0.125 0.111111111111111 0.1 0.09090909090909091 0.08333333333333333 0.07692307692307693 0.07142857142857142 0.06666666666666667 0.0625 0.058823529411764705 0.05555555555555555
0.111111111111111 0.1 0.09090909090909091 0.08333333333333333 0.07692307692307693 0.07142857142857142 0.06666666666666667 0.0625 0.058823529411764705 0.05555555555555555 0.05263157894736842
0.1 0.09090909090909091 0.08333333333333333 0.07692307692307693 0.07142857142857142 0.06666666666666667 0.0625 0.058823529411764705 0.05555555555555555 0.05263157894736842
```

```
== SISTEM PERSAMAAN LINIER ==
Input dari keyboard atau file?
1. Keyboard
2. File
Masukkan pilihanmu: 2
Masukkan nama file: SPL5
Metode yang ingin digunakan?
1. Eliminasi Gauss
2. Eliminasi Gauss-Jordan
3. Matriks Balikan
4. Cramer
Masukkan pilihanmu: 2
Tidak ada Solusi
```

Analisis: Penyelesaian matriks hilbert n = 10 dengan metode gauss jordan menghasilkan nilai null atau tidak ada solusi, ini mungkin disebabkan oleh elemen pada matriks yang sangat kecil sehingga terbaca nol oleh program.

#### 4.1.2 SPL berbentuk matriks augmented

a. Kasus 1

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

Potongan akhir penyelesaian program :

```

Masukkan elemen matriks:
1 -1 2 -1 -1
2 1 -2 -2 -2
-1 2 -4 1 1
3 0 0 -3 -3
Metode yang ingin digunakan?
1. Eliminasi Gauss
2. Eliminasi Gauss-Jordan
3. Matriks Balikan
4. Cramer
Masukkan pilihanmu: 4
SPL tidak memiliki solusi unik karena determinan matriks A = 0.
Masukkan pilihanmu: 2
X1 = -1.0 + X4
X2 = 0.0 + 2.0X3
X3 = free
X4 = free

```

Analisis : masukan berasal dari input keyboard. Dengan menggunakan metode gauss jordan, persamaan diatas dapat diselesaikan, dan menghasilkan solusi parametrik.

b. Kasus 2

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

Pada test case ini, matriks disimpan dalam file bernama SPL7.txt sebagai berikut:

```

1 2 0 8 0 8
2 0 1 0 4 6
3 -4 0 6 0 6
4 0 -2 0 3 -1
5 2 0 -4 0 -4
6 0 1 0 -2 0

```

Potongan akhir penyelesaian program:

```

== SISTEM PERSAMAAN LINIER ==
Input dari keyboard atau file?
1. Keyboard
2. File
Masukkan pilihanmu: 2
Masukkan nama file: SPL7
Metode yang ingin digunakan?
1. Eliminasi Gauss
2. Eliminasi Gauss-Jordan
3. Matriks Balikan
4. Cramer
Masukkan pilihanmu: 1
X1 = 0.0
X2 = 2.0
X3 = 1.0
X4 = 1.0

```

Analisis : Penyelesaian program dilakukan dengan metode gauss jordan menghasilkan solusi tunggal (unik).

#### 4.1.3 SPL dengan kasus

##### a. Kasus 1

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

Pada test case ini, persamaan matriks dimasukkan dengan input keyboard. Berikut potongan penyelesaian program :

```

Masukkan elemen matriks:
8 1 3 2 0
2 9 -1 -2 1
1 3 2 -1 2
1 0 6 4 3
Metode yang ingin digunakan?
1. Eliminasi Gauss
2. Eliminasi Gauss-Jordan
3. Matriks Balikan
4. Cramer
Masukkan pilihanmu: 3
Solusi:
x1 = -0,2243
x2 = 0,1824
x3 = 0,7094
x4 = -0,2582

```

Analisis : persoalan diatas dapat diselesaikan dengan metode matriks balikan karena nilai determinannya tidak nol (solusi unik).

b. Kasus 2

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

Persamaan diatas disederhanakan, dijadikan matriks augmented lalu dimasukkan ke file bernama SPL9.txt

Isi file SPL9.txt :

```
1 0 0 0 0 0 0 1 1 1 13
2 0 0 0 1 1 1 0 0 0 15
3 1 1 1 0 0 0 0 0 0 8
4 0 0 0.04289 0 0.04289 0.75 0.04289 0.75 0.61396 14.79
5 0 0.25 0.91421 0.25 0.91421 0.25 0.91421 0.25 0 14.31
6 0.61396 0.75 0.04289 0.75 0.04289 0 0.04289 0 0 3.81
7 0 0 1 0 0 1 0 0 1 18
8 0 1 0 0 1 0 0 1 0 12
9 1 0 0 1 0 0 1 0 0 6
10 0.04289 0.75 0.61396 0 0.04289 0.75 0 0 0.04289 10.51
11 0.91421 0.25 0 0.25 0.91421 0.25 0 0.25 0.91421 16.13
12 0.04289 0 0 0.75 0.04289 0 0.61396 0.75 0.04289 7.04
```

Potongan penyelesaian program :



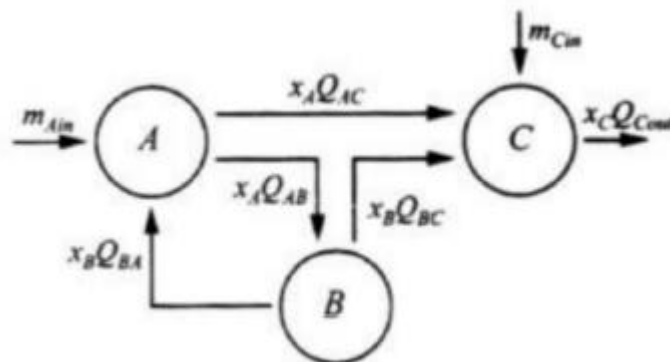
```

== SISTEM PERSAMAAN LINIER ==
Input dari keyboard atau file?
1. Keyboard
2. File
Masukkan pilihanmu: 2
Masukkan nama file: SPL9
Metode yang ingin digunakan?
1. Eliminasi Gauss
2. Eliminasi Gauss-Jordan
3. Matriks Balikan
4. Cramer
Masukkan pilihanmu: 2
Tidak ada Solusi

```

Analisis : dengan menggunakan metode gauss jordan, SPL diatas tidak memiliki solusi.

c. Kasus 3



Gambar diatas adalah diagram sistem reaktor.

Berikut adalah bentuk persamaan dari sistem reaktor tersebut :

$$A: m_{Ain} + Q_{BA}x_B - Q_{AB}x_A - Q_{AC}x_A = 0$$

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

$$C: m_{Cin} + Q_{AC}x_A + Q_{BC}x_B - Q_{Cout}x_C = 0$$

Nilai dari parameter Q yang akan digunakan dalam persamaan ini adalah  $Q_{AB} = 40$ ,  $Q_{AC} = 80$ ,  $Q_{BA} = 60$ ,  $Q_{BC} = 20$  dan  $Q_{Cout} = 150$  m<sup>3</sup>/s dan  $m_{Ain} = 1300$  dan  $m_{Cin} = 200$  mg/s.

Persamaan diatas dijadikan matriks lalu dimasukkan ke file bernama SPL10.txt

```

1  -120 60 0 -1300
2   40 -80 0 0
3   80 20 -150 -200

```

Kolom pertama adalah koefisien dari  $X_a$ , kolom kedua adalah koefisien dari  $X_b$ , kolom ketiga adalah koefisien dari  $X_c$ , dan kolom keempat adalah nilai  $m$  (konstanta).

Potongan penyelesaian program :

```

== SISTEM PERSAMAAN LINIER ==
Input dari keyboard atau file?
1. Keyboard
2. File
Masukkan pilihanmu: 2
Masukkan nama file: SPL10
Metode yang ingin digunakan?
1. Eliminasi Gauss
2. Eliminasi Gauss-Jordan
3. Matriks Balikan
4. Cramer
Masukkan pilihanmu: 4
Solusi:
x1 = 14,4444
x2 = 7,2222
x3 = 10,0000

```

Analisis : penyelesaian persoalan dilakukan dengan menggunakan kaidah cramer menghasilkan solusi unik (tunggal).

## 4.2 Determinan Matriks

### 4.2.1 Metode Reduksi Baris

Akan dilakukan beberapa eksperimen determinan matriks dengan berbagai macam ukuran dan input.

#### a. Kasus 1x1

```

== DETERMINAN ==
Input dari keyboard atau file?
1. Keyboard
2. File
Masukkan pilihanmu: 1
Masukkan ukuran matriks (n x n): 1
Masukkan elemen matriks (pisahkan dengan spasi):
7

Metode yang ingin digunakan?
1. Ekspansi kofaktor
2. Reduksi baris
Masukkan pilihanmu: 2
Determinan: 7.0

```

b. Kasus 2x2

```

== DETERMINAN ==
Input dari keyboard atau file?
1. Keyboard
2. File
Masukkan pilihanmu: 1
Masukkan ukuran matriks (n x n): 2
Masukkan elemen matriks (pisahkan dengan spasi):
1 3
4 2

Metode yang ingin digunakan?
1. Ekspansi kofaktor
2. Reduksi baris
Masukkan pilihanmu: 2
Determinan: -10.0

```

c. Kasus 3x3

```

== DETERMINAN ==
Input dari keyboard atau file?
1. Keyboard
2. File
Masukkan pilihanmu: 1
Masukkan ukuran matriks (n x n): 3
Masukkan elemen matriks (pisahkan dengan spasi):
0 8 1
4 4 6
-10 -7 3

Metode yang ingin digunakan?
1. Ekspansi kofaktor
2. Reduksi baris
Masukkan pilihanmu: 2
Determinan: -564.0

```

d. Kasus Determinan 0

```

== DETERMINAN ==
Input dari keyboard atau file?
1. Keyboard
2. File
Masukkan pilihanmu: 1
Masukkan ukuran matriks (n x n): 4
Masukkan elemen matriks (pisahkan dengan spasi):
1 -1 2 -1
2 1 -2 -2
-1 2 -4 1
3 0 0 -3

Metode yang ingin digunakan?
1. Ekspansi kofaktor
2. Reduksi baris
Masukkan pilihanmu: 2
Determinan: 0.0

```

e. Kasus Matriks Identitas

```

== DETERMINAN ==
Input dari keyboard atau file?
1. Keyboard
2. File
Masukkan pilihanmu: 1
Masukkan ukuran matriks (n x n): 3
Masukkan elemen matriks (pisahkan dengan spasi):
1 0 0
0 1 0
0 0 1

Metode yang ingin digunakan?
1. Ekspansi kofaktor
2. Reduksi baris
Masukkan pilihanmu: 2
Determinan: 1.0

```

#### 4.2.2 Metode Ekspansi Kofaktor

Akan dilakukan beberapa eksperimen determinan matriks dengan berbagai macam ukuran dan input.

a. Kasus 1x1

```

== DETERMINAN ==
Input dari keyboard atau file?
1. Keyboard
2. File
Masukkan pilihanmu: 1
Masukkan ukuran matriks (n x n): 1
Masukkan elemen matriks (pisahkan dengan spasi):
5

Metode yang ingin digunakan?
1. Ekspansi kofaktor
2. Reduksi baris
Masukkan pilihanmu: 1
Determinan: 5.0

```

b. Kasus 2x2

```
== DETERMINAN ==
Input dari keyboard atau file?
1. Keyboard
2. File
Masukkan pilihanmu: 1
Masukkan ukuran matriks (n x n): 2
Masukkan elemen matriks (pisahkan dengan spasi):
-100 -99
20 50

Metode yang ingin digunakan?
1. Ekspansi kofaktor
2. Reduksi baris
Masukkan pilihanmu: 2
Determinan: -3020.0
```

c. Kasus 3x3

```
== DETERMINAN ==
Input dari keyboard atau file?
1. Keyboard
2. File
Masukkan pilihanmu: 1
Masukkan ukuran matriks (n x n): 3
Masukkan elemen matriks (pisahkan dengan spasi):
4.3 2.5 6.0
4.1 5.3 9.1
0.9 7.5 1.2

Metode yang ingin digunakan?
1. Ekspansi kofaktor
2. Reduksi baris
Masukkan pilihanmu: 1
Determinan: -102.072
```

d. Kasus Determinan 0

```
== DETERMINAN ==
Input dari keyboard atau file?
1. Keyboard
2. File
Masukkan pilihanmu: 1
Masukkan ukuran matriks (n x n): 2
Masukkan elemen matriks (pisahkan dengan spasi):
1 2
1 2

Metode yang ingin digunakan?
1. Ekspansi kofaktor
2. Reduksi baris
Masukkan pilihanmu: 1
Determinan: 0.0
```

e. Kasus Matriks Identitas

```

== DETERMINAN ==
Input dari keyboard atau file?
1. Keyboard
2. File
Masukkan pilihanmu: 1
Masukkan ukuran matriks (n x n): 2
Masukkan elemen matriks (pisahkan dengan spasi):
1 0
0 1

Metode yang ingin digunakan?
1. Ekspansi kofaktor
2. Reduksi baris
Masukkan pilihanmu: 1
Determinan: 1.0

```

### 4.3 Balikan Matriks

#### 4.3.1 Metode Gauss-Jordan

Akan dilakukan beberapa eksperimen invers matriks dengan berbagai macam ukuran dan input.

##### f. Kasus 1x1

```

== INVERS ATAU MATRIKS BALIKAN ==
Input dari keyboard atau file?
1. Keyboard
2. File
Masukkan pilihanmu: 1
Masukkan ukuran matriks (n x n): 1
Masukkan elemen matriks (pisahkan dengan spasi):
5

Metode yang ingin digunakan?
1. Invers Gauss-Jordan
2. Invers Adjoin
Masukkan pilihanmu: 1
0.2000

```

##### g. Kasus 2x2

```

== INVERS ATAU MATRIKS BALIKAN ==
Input dari keyboard atau file?
1. Keyboard
2. File
Masukkan pilihanmu: 1
Masukkan ukuran matriks (n x n): 2
Masukkan elemen matriks (pisahkan dengan spasi):
3 4
2 6

Metode yang ingin digunakan?
1. Invers Gauss-Jordan
2. Invers Adjoin
Masukkan pilihanmu: 1
0.6000 -0.4000
-0.2000 0.3000

```

h. Kasus 3x3

```
== INVERS ATAU MATRIKS BALIKAN ==
Input dari keyboard atau file?
1. Keyboard
2. File
Masukkan pilihanmu: 1
Masukkan ukuran matriks (n x n): 3
Masukkan elemen matriks (pisahkan dengan spasi):
4 5 9
1 3 5
2 5 1

Metode yang ingin digunakan?
1. Invers Gauss-Jordan
2. Invers Adjoin
Masukkan pilihanmu: 1
0.4231 -0.7692 0.0385
-0.1731 0.2692 0.2115
0.0192 0.1923 -0.1346
```

i. Kasus Matriks Singular

```
== INVERS ATAU MATRIKS BALIKAN ==
Input dari keyboard atau file?
1. Keyboard
2. File
Masukkan pilihanmu: 1
Masukkan ukuran matriks (n x n): 3
Masukkan elemen matriks (pisahkan dengan spasi):
1 2 3
4 5 6
7 8 9

Metode yang ingin digunakan?
1. Invers Gauss-Jordan
2. Invers Adjoin
Masukkan pilihanmu: 1
Matrix tidak memiliki invers (singular).
```

j. Kasus Matriks Identitas

```
== INVERS ATAU MATRIKS BALIKAN ==
Input dari keyboard atau file?
1. Keyboard
2. File
Masukkan pilihanmu: 1
Masukkan ukuran matriks (n x n): 3
Masukkan elemen matriks (pisahkan dengan spasi):
1 0 0
0 1 0
0 0 1

Metode yang ingin digunakan?
1. Invers Gauss-Jordan
2. Invers Adjoin
Masukkan pilihanmu: 1
1.0000 0.0000 0.0000
0.0000 1.0000 0.0000
0.0000 0.0000 1.0000
```

#### 4.3.2 Metode Adjoin

Akan dilakukan beberapa eksperimen invers matriks dengan berbagai macam ukuran dan input.

a. Kasus 1x1

```
== INVERS ATAU MATRIKS BALIKAN ==  
Input dari keyboard atau file?  
1. Keyboard  
2. File  
Masukkan pilihanmu: 1  
Masukkan ukuran matriks (n x n): 1  
Masukkan elemen matriks (pisahkan dengan spasi):  
7  
  
Metode yang ingin digunakan?  
1. Invers Gauss-Jordan  
2. Invers Adjoin  
Masukkan pilihanmu: 2  
0.1429
```

b. Kasus 2x2

```
== INVERS ATAU MATRIKS BALIKAN ==  
Input dari keyboard atau file?  
1. Keyboard  
2. File  
Masukkan pilihanmu: 1  
Masukkan ukuran matriks (n x n): 2  
Masukkan elemen matriks (pisahkan dengan spasi):  
5 -2  
-1 0  
  
Metode yang ingin digunakan?  
1. Invers Gauss-Jordan  
2. Invers Adjoin  
Masukkan pilihanmu: 2  
0.0000 -1.0000  
-0.5000 -2.5000
```

c. Kasus 3x3



```

== INVERS ATAU MATRIKS BALIKAN ==
Input dari keyboard atau file?
1. Keyboard
2. File
Masukkan pilihanmu: 1
Masukkan ukuran matriks (n x n): 3
Masukkan elemen matriks (pisahkan dengan spasi):
5 3 -1
-10 7 8
0 1 2

Metode yang ingin digunakan?
1. Invers Gauss-Jordan
2. Invers Adjoin
Masukkan pilihanmu: 2
0.0600 -0.0700 0.3100
0.2000 0.1000 -0.3000
-0.1000 -0.0500 0.6500

```

d. Kasus Matriks Singular

```

== INVERS ATAU MATRIKS BALIKAN ==
Input dari keyboard atau file?
1. Keyboard
2. File
Masukkan pilihanmu: 1
Masukkan ukuran matriks (n x n): 4
Masukkan elemen matriks (pisahkan dengan spasi):
1 2 3 4
5 6 7 8
9 10 11 12
13 14 15 16

Metode yang ingin digunakan?
1. Invers Gauss-Jordan
2. Invers Adjoin
Masukkan pilihanmu: 2
Matrix tidak dapat dibalik, determinan = 0.

```

e. Kasus Matriks Identitas

```

== INVERS ATAU MATRIKS BALIKAN ==
Input dari keyboard atau file?
1. Keyboard
2. File
Masukkan pilihanmu: 1
Masukkan ukuran matriks (n x n): 2
Masukkan elemen matriks (pisahkan dengan spasi):
1 0
0 1

Metode yang ingin digunakan?
1. Invers Gauss-Jordan
2. Invers Adjoin
Masukkan pilihanmu: 2
1.0000 0.0000
0.0000 1.0000

```

## 4.4 Interpolasi

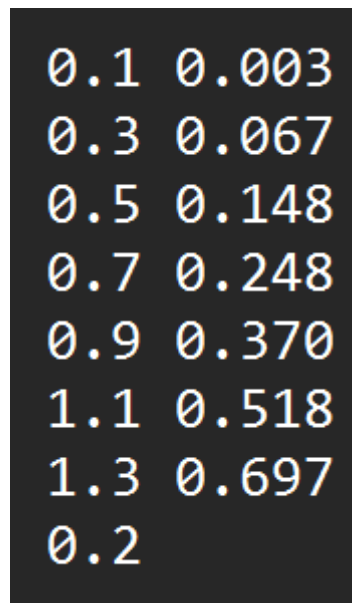
#### 4.4.1 Interpolasi Polinom

##### a. Kasus 1

Akan dilakukan interpolasi terhadap titik-titik berikut untuk menaksir nilai-nilai pada titik-titik yaitu  $x = 0.2$ ,  $x = 0.55$ ,  $x = 0.85$ , dan  $x = 1.28$ :

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

Titik-titik tersebut dituliskan dalam sebuah text file dalam bentuk sebagai berikut, dengan enam baris pertama adalah data yang tersedia dan baris terakhir adalah nilai yang ingin ditaksir.



```
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
0.2
```

Gambar 4.4.2.1  
Isi dari file studi\_kasus.txt

Dilakukan interpolasi lalu ditemukan taksiran-taksiran sebagai berikut

```
= INTERPOLASI POLINOM =
Input dari keyboard atau file?
1. Keyboard
2. File
Masukkan pilihanmu: 2
Masukkan nama file (tanpa path): studi_kasus
y = -0,0230 + 0,2400 * x^1 + 0,1974 * x^2 - 0,0000 * x^3 + 0,0260 * x^4 + 0,0000 * x^5 + 0,0000 * x^6
f(0.2) =0.032960937499999995

y = -0,0230 + 0,2400 * x^1 + 0,1974 * x^2 - 0,0000 * x^3 + 0,0260 * x^4 + 0,0000 * x^5 + 0,0000 * x^6
f(0.55) =0.17111865234375

y = -0,0230 + 0,2400 * x^1 + 0,1974 * x^2 - 0,0000 * x^3 + 0,0260 * x^4 + 0,0000 * x^5 + 0,0000 * x^6
f(0.85) =0.33723583984374994

y = -0,0230 + 0,2400 * x^1 + 0,1974 * x^2 - 0,0000 * x^3 + 0,0260 * x^4 + 0,0000 * x^5 + 0,0000 * x^6
f(1.28) =0.6775418374999999
```

## Analisis

### b. Kasus 2

Akan dilakukan interpolasi terhadap data jumlah kasus positif baru Covid-19 di Indonesia dari tanggal 17 Juni 2022 sampai 31 Agustus 2022. Tanggal telah diolah menjadi desimal dengan perhitungan: Tanggal (desimal) = bulan + (tanggal / jumlah hari pada bulan tersebut).

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

Akan ditaksir jumlah kasus baru pada tanggal-tanggal berikut:

Tanggal	Tanggal (Desimal)
16/07/2022	7.51612903226
10/08/2022	8.32258064516
05/09/2022	9.16666666667
06/10/2022	10.1935483871

Titik-titik tersebut dituliskan dalam sebuah text file dalam bentuk sebagai berikut, dengan enam baris pertama adalah data yang tersedia dan baris terakhir adalah nilai yang ingin ditaksir.

```

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
7.51612903226

```

Dilakukan interpolasi dan ditemukan taksiran-taksiran sebagai berikut

```

== INTERPOLASI POLINOM ==
Input dari keyboard atau file?
1. Keyboard
2. File
Masukkan pilihanmu: 2
Masukkan nama file (tanpa path): studi_kasus_2
y = 7186396850227,1410 - 9346209175934,1100 * x^1 + 5333795523226,5530 * x^2 - 1756686771893,4155 * x^3 + 36851935483871
f(7.51612903226) =53532.587890625

y = 7186396850227,1410 - 9346209175934,1100 * x^1 + 5333795523226,5530 * x^2 - 1756686771893,4155 * x^3 + 36851935483871
f(8.32258064516) =36315.6328125

y = 7186396850227,1410 - 9346209175934,1100 * x^1 + 5333795523226,5530 * x^2 - 1756686771893,4155 * x^3 + 36851935483871
f(9.16666666667) =-664780.5859375

y = 7186396850227,1410 - 9346209175934,1100 * x^1 + 5333795523226,5530 * x^2 - 1756686771893,4155 * x^3 + 36851935483871
f(10.1935483871) =-4.9769353425E8

```

c. Kasus 3

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

Akan dilakukan penyederhanaan terhadap fungsi tersebut dengan mengambil titik-titik x di dalam selang [0,2] sebanyak 8 titik. Titik- titik yang diambil adalah

x	f(x)
0	0
0.25	0.36668232
0.5	0.44543086
0.75	0.49826487
1	0.53788284
1.25	0.56547258
1.5	0.580869693
1.75	0.58435813
2	0.57665152

Setelah dilakukan interpolasi, diperoleh hasil sebagai berikut.

```

== INTERPOLASI POLINOM ==
Input dari keyboard atau file?
1. Keyboard
2. File
Masukkan pilihanmu: 2
Masukkan nama file (tanpa path): studi_kasus_3
y = 0,0000 + 3,2150 * x^1 - 11,5309 * x^2 + 24,7332 * x^3 - 31,9435 * x^4 + 25,2735 * x^5 - 11,9971 * x^6 + 3,1336 * x^7 - 0,3458 * x^8

```

#### 4.4.2 Interpolasi Bicubic Spline

Akan diselesaikan studi kasus interpolasi bicubic spline dengan masukan data berbentuk matriks seperti berikut :

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

Eksperimen akan dibagi menjadi 2 berdasarkan metode inputnya, masukan dari file dan masukan dari keyboard. Masukan dari file ada 2 test case dengan  $a = 0$ ,  $b = 0$ , dan  $a = 0.5$ ,  $b = 0.5$ . sedangkan untuk masukan dari keyboard ada 2 test case juga yaitu  $a = 0.25$ ,  $b = 0.75$  dan  $a = 0.1$ ,  $b = 0.9$

##### 4.4.2.1 Masukan dari File

- a. Solusi untuk  $a = 0$ , dan  $b = 0$

Test case berada di file bernama BS1.txt

```

1  21 98 125 153
2  51 101 161 59
3  0 42 72 210
4  16 12 81 96
5  0 0

```

```

Hak cipta sepenuhnya dimiliki oleh SadangSerang Co.
Selamat datang di program pemecah persamaan dan penghitung regresi!
1. Sistem Persamaan Linier
2. Determinan
3. Matriks Balikan
4. Interpolasi Polinom
5. Interpolasi Bicubic Spline
6. Regresi Linier
7. Regresi Kuadratik Berganda
8. Interpolasi Gambar
9. Keluar
Masukkan pilihanmu: 5
Input dari keyboard atau file?
1. Keyboard
2. File
Masukkan pilihanmu: 2
Masukkan nama file (tanpa path): BS1.txt
Hasil interpolasi : 21.0

```

Analisis : Program penyelesaian Interpolasi Bicubic Spline menghasilkan jawaban 21

- b. Solusi untuk  $a = 0.5$  dan  $b = 0.5$   
 Test case berada di file bernama BS2.txt

```

1  21 98 125 153
2  51 101 161 59
3  0 42 72 210
4  16 12 81 96
5  0.5 0.5

```

```

Hak cipta sepenuhnya dimiliki oleh SadangSerang Co.
Selamat datang di program pemecah persamaan dan penghitung regresi!
1. Sistem Persamaan Linier
2. Determinan
3. Matriks Balikan
4. Interpolasi Polinom
5. Interpolasi Bicubic Spline
6. Regresi Linier
7. Regresi Kuadratik Berganda
8. Interpolasi Gambar
9. Keluar
Masukkan pilihanmu: 5
Input dari keyboard atau file?
1. Keyboard
2. File
Masukkan pilihanmu: 2
Masukkan nama file (tanpa path): BS2.txt
Hasil interpolasi : 87.796875

```

#### 4.4.2.2 Masukan dari Keyboard

- a. Solusi untuk  $a = 0.25$  dan  $b = 0.75$

```

== INTERPOLASI BICUBIC SPLINE ==
Input dari keyboard atau file?
1. Keyboard
2. File
Masukkan pilihanmu: 1
Masukkan jumlah baris: 16
Masukkan jumlah kolom: 1
Masukkan elemen matriks:
21
98
125
153
51
101
161
59
0
42
72
210
16
12
81
96
Masukkan nilai a : 0.25
Masukkan nilai b : 0.75
Hasil interpolasi : 117.732177734375

```

- b. Solusi untuk  $a = 0.1$  dan  $b = 0.9$

```

== INTERPOLASI BICUBIC SPLINE ==
Input dari keyboard atau file?
1. Keyboard
2. File
Masukkan pilihanmu: 1
Masukkan jumlah baris: 16
Masukkan jumlah kolom: 1
Masukkan elemen matriks:
21
98
125
153
51
101
161
59
0
42
72
210
16
12
81
96
Masukkan nilai a : 0.1
Masukkan nilai b : 0.9
Hasil interpolasi : 128.57518700000003

```

#### 4.5 Regresi Linier Berganda

Akan diselesaikan studi kasus nomor 6 yang diberikan data sebagai berikut.

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.

Pada soal juga diminta untuk mengestimasi nilai NO apabila Humidity bernilai 50%, temperature 76°F, dan tekanan udara sebesar 29.30.

Dari data diatas, didapatkan solusi sebagai berikut.



```

== REGRESI LINIER BERGANDA ==
Input dari keyboard atau file?
1. Keyboard
2. File
Masukkan pilihanmu: 2
Masukkan nama file: studi_kasus
y = -3,5078 + -0,0026 * x1 + 0,0008 * x2 + 0,1542 * x3
f(50.0, 76.0, 29.3) = 0.9384342262216845

```

Analisis:

Data yang diberikan akan dimasukkan ke dalam program dalam bentuk matriks augmented yang akan dibuat menjadi bentuk Normal Estimation Equation for Multiple Linear Regression. Setelah dalam bentuk tersebut, persamaan akan dipecahkan menggunakan package sistem persamaan linier yang ada. Lalu, program akan mengestimasi nilai NO berdasarkan nilai Humidity, temperature, dan juga tekanan udara yang diberikan. Dapat dilihat pada gambar di atas, program memprediksi nilai NO yaitu sebesar 0.938434. Nilai yang diperoleh tidak jauh berbeda dengan regresi kuadratik berganda.

#### 4.6 Regresi Kuadratik Berganda

Akan diselesaikan studi kasus nomor 6 yang diberikan data sebagai berikut.

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.

Pada soal juga diminta untuk mengestimasi nilai NO apabila Humidity bernilai 50%, temperature 76°F, dan tekanan udara sebesar 29.30.

Dari data diatas, didapatkan solusi sebagai berikut.

```

Persamaan regresi kuadratik:
y = -1146.4371 + 0.1840 * x1 + 0.8386 * x2 + 75.4503 * x3 + -0.0000 * x1^2 + -0.0002 * x2^2 + -1.2404
* x3^2 + 0.0000 * x1 * x2 + -0.0064 * x1 * x3 + -0.0272 * x2 * x3
Prediksi nilai untuk observasi [50.0, 76.0, 29.3] adalah: 0.9439

```

Analisis:

Data yang diberikan kemudian dimasukkan ke dalam program dalam bentuk Matriks Augmented. Program kemudian akan melakukan training dari data yang telah diberikan. Setelah itu, dengan menggunakan metode Ordinary Least Square (OLS) didapatkan nilai koefisien atau beta dari setiap variabel. Kemudian, akan terbentuk persamaan regresi kuadratik berganda seperti yang terdapat pada gambar di atas. Lalu, program akan mengestimasi nilai NO berdasarkan nilai Humidity, temperature, dan juga tekanan udara yang diberikan. Dapat dilihat pada gambar di atas, program memprediksi nilai NO yaitu sebesar 0.9439.

#### 4.7 Image Resizing

Dilakukan image resizing terhadap sebuah gambar berukuran 680x383 dengan *size* 28,8kb. Gambar tersebut diubah menjadi dua kali lipat untuk width dan heightnya, menjadi 1366x760 dengan *size* 220kb. (Gambar kiri adalah sebelum, gambar kanan adalah sesudah.)



Analisis:

Analisis dilakukan dengan memecah pixel-pixel dalam gambar menjadi bentuk-bentuk dan juga mengambil data RGB. Lalu, tiap-tiap grid tersebut dikalikan dengan matriks image resizing yang berasal dari sejumlah turunan berarah. Diperoleh sebuah gambar yang berukuran lebih besar dibanding gambar semula.

## 5 KESIMPULAN

### 5.1 Kesimpulan

Berbagai persoalan dalam matematika bisa diselesaikan dengan program komputer, salah satunya adalah Sistem Persamaan Linear. Dengan adanya program penyelesaian Sistem Persamaan Linear, banyak persoalan dapat diselesaikan dengan mudah seperti perhitungan interpolasi, pemrosesan citra, perhitungan regresi, dan persoalan matematis lainnya.

### 5.2 Saran

Penulis ingin menyampaikan apresiasi yang sebesar-besarnya kepada para asisten mata kuliah atas bimbingan dan dukungannya selama proses pengerjaan tugas ini. Masukan serta arahan yang diberikan sangat membantu dalam menyelesaikan tugas ini dengan baik. Namun, penulis juga menyarankan agar penyusunan spesifikasi tugas di masa mendatang dapat lebih rinci dan jelas, khususnya dalam menjelaskan setiap poin yang harus dipenuhi. Dengan penjelasan yang lebih terperinci, diharapkan mahasiswa dapat lebih mudah memahami dan mengikuti instruksi yang diberikan, sehingga hasil akhir yang dicapai bisa lebih optimal.

### **5.3 Komentar**

Tugas besar ini memberikan banyak pengalaman pelajaran bagi kami. Dengan adanya tugas besar ini, kami jadi tahu bagaimana memogram dengan bahasa java, bagaimana memrogram persoalan matematika di komputer, dan juga bagaimana mekanisme pengolahan citra gambar dengan persamaan matematis.

### **5.4 Refleksi**

Pengerjaan Tugas Besar sebaiknya dilakukan dan diselesaikan dari jauh – jauh hari untuk menghindari berbagai macam kendala yang tidak terduga. Analisis kebutuhan dan analisis dependensi juga perlu di diskusikan sebelum memulai pengembangan program.

## **6 LAMPIRAN**

### **6.1 Referensi**

<https://math.stackexchange.com/questions/3155866/multivariate-quadratic-regression>

[How to Build Your Own Java library? – Program Creek](#)

### **6.2 Tautan Repository Github**

<https://github.com/mraifalkautsar/Algeo01-23011>

### **6.3 Tautan Video**

<https://youtu.be/e5Y-tWbcZ9c?>