

**Laporan Tugas Besar 1**  
**IF2123 ALJABAR LINEAR DAN GEOMETRI**  
Sistem Persamaan Linear, Matriks, dan Aplikasinya



Disusun oleh:

Muhammad Atpur Rafif (13522086)  
M. Hanief Fatkhan Nashrullah (13522100)  
Indraswara Galih Jayanegara (13522119)

**Program Studi Teknik Informatika  
Sekolah Teknik Elektro dan Informatika  
Institut Teknologi Bandung  
2022**

## Table of Contents

BAB I .....	3
Tujuan.....	3
Spesifikasi Program.....	3
BAB II.....	5
1. Operasi Matrix.....	5
2. Interpolasi Polinomial.....	9
3. Regresi Berganda.....	10
4. Bicubic Interpolation .....	10
BAB III .....	12
BAB IV .....	13
BAB V .....	19

# BAB I

## Tujuan

Tujuan dari tugas besar 1 mata kuliah Aljabar Linear dan Geometri ini adalah:

1. Membuat pustaka (library atau package) dalam Bahasa Java untuk menemukan solusi SPL dengan metode eliminasi Gauss, metode Eliminasi Gauss-Jordan, metode matriks balikan, dan kaidah Cramer (kaidah Cramer khusus untuk SPL dengan n peubah dan n persamaan), menghitung determinan matriks dengan reduksi baris dan dengan ekspansi kofaktor, dan menghitung balikan matriks.
2. Menggunakan pustaka yang sudah dibuat untuk membuat program penyelesaian berbagai persoalan dalam bentuk SPL, dan menyelesaikan persoalan interpolasi dan regresi linier, menghitung matriks balikan, menghitung determinan matriks dengan berbagai metode (reduksi baris dan ekspansi kofaktor).

## Spesifikasi Program

1. Program dapat menerima masukan (*input*) baik dari *keyboard* maupun membaca masukan dari *file text*. Untuk SPL, masukan dari keyboard adalah  $m, n$ , koefisian  $a_{ij}$ ,  $b$ . Masukan dari *file* berbentuk matrix *augmented* tanpa tanda kurung. Setiap elemen matrix dipisah oleh spasi. Misalnya,

$$\begin{array}{ccccc} 3 & 4.5 & 2.8 & 10 & 12 \\ -3 & 7 & 8.3 & 11 & -4 \\ 0.5 & -10 & -9 & 12 & 0 \end{array}$$

2. Untuk persoalan menghitung determinan dan matrix balikan, masukan dari *keyboard* adalah  $n$  dan koefisian  $a_{ij}$ . Masukan dari *file* berbentuk matrix, setiap elemen matrix dipisah oleh spasi. Misalnya,

$$\begin{array}{ccccc} 3 & 4.5 & 2.8 & & \\ -3 & 7 & 8.3 & & \\ 0.5 & -10 & -9 & & \end{array}$$

3. Untuk persoalan interpolasi, masukannya jika dari *keyboard* adalah  $n, (x_0, y_0), (x_1, y_1), \dots, (x_n, y_n)$ , dan nilai  $x$  yang akan ditaksir nilai fungsinya. Jika masukannya dari *file*, maka titik-titik dinyatakan pada setiap baris tanpa koma dan tanda kurung. Masukan kemudian dilanjutkan dengan satu buah baris berisi satu buah nilai  $x$  yang akan ditaksir menggunakan fungsi interpolasi yang telah didefinisikan. Misalnya jika titik-titik datanya adalah  $(8.0, 2.0794), (9.0, 2.1927)$ , dan  $(9.5, 2.2513)$  dan akan mencari nilai  $y$  saat  $x = 8.3$ , maka di dalam *file text* ditulis sebagai berikut:

$$\begin{array}{cc} 8.0 & 2.0794 \\ 9.0 & 2.1927 \\ 9.5 & 2.2513 \\ 8.3 & \end{array}$$

4. Untuk persoalan regresi, masukannya jika dari *keyboard* adalah  $n$  (jumlah peubah  $x$ ),  $m$  (jumlah sampel), semua nilai-nilai  $x_{1i}, x_{2i}, \dots, x_{ni}$ , nilai  $y_i$ , dan nilai-nilai  $x_k$  yang akan ditaksir nilai fungsinya. Jika masukannya dari *file*, maka titik-titik dinyatakan pada setiap baris tanpa koma dan tanda kurung.
5. Untuk persoalan SPL, luaran program adalah solusi SPL. Jika solusinya tunggal, tuliskan nilainya. Jika solusinya tidak ada, tuliskan solusi tidak ada. Jika solusinya banyak, maka

tuliskan solusinya dalam bentuk parametrik (misalnya  $x_4 = -2$ ,  $x_3 = 2s - t$ ,  $x_2 = s$ , dan  $x_1 = t$ )

6. Untuk persoalan polinom interpolasi dan regresi, luarannya adalah persamaan polinom/regresi dan taksiran nilai fungsi pada x yang diberikan. Contoh luaran untuk interpolasi adalah

$$f(x) = -0.0064x^2 + 0.2266x + 0.6762, f(5) = \dots$$

dan untuk regresi adalah

$$f(x) = -9.5872 + 1.0732x_1, f(x_k) = \dots$$

7. Untuk persoalan *bicubic spline interpolation*, masukan dari *file text* (.txt) yang berisi matriks berukuran 4 x 4 yang berisi konfigurasi nilai fungsi dan turunan berarah disekitarnya, diikuti dengan nilai  $a$  dan  $b$  untuk mencari nilai  $f(a, b)$ . Misalnya jika nilai dari  $f(0, 0), f(1, 0), f(0, 1), f(1, 1), f_{xx}(0, 0), f_{xx}(1, 0), f_{xx}(0, 1), f_{xx}(1, 1), f_{xy}(0, 0), f_{xy}(1, 0), f_{xy}(0, 1), f_{xy}(1, 1)$  berturut-turut adalah 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16 serta nilai  $a$  dan  $b$  yang dicari berturut-turut adalah 0.5 dan 0.5 maka isi *file text* ditulis sebagai berikut:

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16
0.5	0.5		

Luaran yang dihasilkan adalah nilai dari  $f(0.5, 0.5)$ .

8. Luaran program harus dapat ditampilkan **pada layar komputer dan dapat disimpan ke dalam file**.
9. Bahasa program yang digunakan adalah Java. Anda bebas untuk menggunakan versi java apapun dengan catatan di atas java versi 8 (8/9/11/15/17/19/20).
10. Program **tidak harus** berbasis GUI, cukup *text-based* saja, namun boleh menggunakan GUI (memakai kakas *Eclipse* misalnya).
11. Program dapat dibuat dengan pilihan menu. Urutan menu dan isinya dipersilakan dirancang masing-masing. Misalnya, menu:

1. Sistem Persamaan Linier
2. Determinan
3. Matriks balikan
4. Interpolasi Polinom
5. Interpolasi Bicubic Spline
6. Regresi linier berganda
7. Keluar

Untuk pilihan menu nomor 1 ada sub-menu lagi yaitu pilihan metode:

1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Metode matriks balikan
4. Kaidah Cramer

Begitu juga untuk pilihan menu nomor 2 dan 3.

## BAB II

### 1. Operasi Matrix

#### a. Determinant Matriks

##### i. Metode Ekspansi Kofaktor

Jika X adalah matriks persegi, maka minor dari  $a_{ij}$  dinyatakan dengan  $M_{ij}$  dan ditafsirkan menjadi determinan submatriks yang tetap setelah baris ke- $i$  dan kolom ke- $j$  dicoret. Menjadi determinan submatriks yang tetap setelah baris ke- $i$  dan kolom ke- $j$  dicoret dari A. kofaktor sendiri bernilai  $(-1)^{i+j} M_{ij}$  yang dinyatakan dengan  $C_{ij}$ .

Berikut contoh dari kofaktor dan minor.

$$M = \begin{bmatrix} 0 & 3 & 6 \\ 1 & 5 & -2 \\ 1 & 2 & 4 \end{bmatrix}$$

Pertama, jika terdapat baris atau kolom dengan angka nol terbanyak, gunakan baris atau kolom tersebut untuk melakukan ekspansi kofaktor. Dalam contoh ini, digunakan baris pertama untuk ekspansi kofaktor.

$$M = \begin{bmatrix} \textcolor{blue}{0} & \textcolor{blue}{3} & \textcolor{blue}{6} \\ 1 & 5 & -2 \\ 1 & 2 & 4 \end{bmatrix}$$

Sehingga kofaktornya adalah

$$\det(M) = (0 \times \begin{vmatrix} 5 & -2 \\ 2 & 4 \end{vmatrix}) - (3 \times \begin{vmatrix} 1 & -2 \\ 1 & 4 \end{vmatrix}) + (6 \times \begin{vmatrix} 1 & 5 \\ 1 & 2 \end{vmatrix})$$

##### ii. Metode Reduksi Baris

Determinan juga dapat dicari dengan melakukan Operasi Baris Elementer atau OBE untuk mendapatkan segitiga atas atau bawah. Poin utama dari metode ini adalah menjadikan sebuah matriks menjadi matriks segitiga atas atau bawah lalu dihitung perkalian dari diagonal utama matriksnya.

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

$p$ : menyatakan banyak operasi pertukaran baris di dalam OBE

$k_1, k_2, \dots, k_m$ : menyatakan perkalian baris-baris OBE

Berikut contoh pencarian determinan matriks dengan metode reduksi baris

$$M = \begin{bmatrix} 0 & 3 & 6 \\ 1 & 5 & -2 \\ 1 & 2 & 4 \end{bmatrix}$$

Pada matriks tersebut, terdapat angka 0 di diagonal utama, maka baris pertama perlu ditukar dengan baris lain. Pada contoh ini, dilakukan penukaran pada baris pertama dengan baris ketiga

$$\begin{bmatrix} 1 & 2 & 4 \\ 1 & 5 & -2 \\ 0 & 3 & 6 \end{bmatrix}$$

Untuk mendapatkan matriks segitiga atas, elemen di bawah *leading one* perlu dibuat menjadi nol dengan OBE. Untuk membuat elemen di bawah *leading one* baris pertama menjadi nol, baris kedua perlu dikurangi dengan baris pertama.

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

Setelah itu, baris ketiga dikurangi dengan baris kedua.

$$\begin{bmatrix} 1 & 2 & 4 \\ 0 & 3 & -6 \\ 0 & 0 & 12 \end{bmatrix}$$

Karena sudah didapatkan matriks segitiga atas, maka determinan dapat dihitung dengan persamaan :

$$\det(M) = \frac{(-1)^p a'_{11} a'_{22} \dots a'_{nn}}{k_1 k_2 k_3}$$

Dalam contoh ini, pertukaran baris dilakukan sebanyak satu kali, dan tidak ada perkalian baris dengan konstanta atau bisa juga dianggap tiap baris dikali dengan satu. Maka determinan matriksnya adalah:

$$\begin{aligned} \det(M) &= \frac{(-1)^1 1.3.12}{1} \\ &= -36 \end{aligned}$$

### b. Metode Eliminasi Gauss

Metode eliminasi gauss berhubungan dengan matriks eselon baris karena eliminasi gauss menjadikan sebuah matriks berbentuk matriks eselon baris.

Matriks Eselon Baris memiliki beberapa sifat berikut:

1. Jika sebuah baris tidak terdiri dari seluruhnya 0, maka bilangan bukan 0 pertama dalam baris tersebut adalah 1 (*leading one*).
2. Jika ada baris yang seluruhnya 0, maka baris tersebut harus diposisikan pada bagian paling bawah pada matriks.
3. Pada dua baris berurutan yang tidak seluruhnya 0, maka 1 utama (*leading one*) pada baris yang lebih rendah terletak lebih kanan daripada 1 utama pada baris yang lebih tinggi.

Berikut contoh penyelesaian Sistem Persamaan Linear dengan menggunakan Metode Eliminasi Gauss. Pada permasalahan di bawah ini, baris pertama memiliki *leading one* pada elemen baris pertama kolom pertama.

$$\left[ \begin{array}{ccc|c} 1 & 1 & -1 & -2 \\ 2 & -1 & 1 & 5 \\ -1 & 2 & 2 & 1 \end{array} \right]$$

Untuk *leading one* baris kedua, posisi kolomnya harus lebih rendah dibandingkan dengan *leading one* baris pertama. OBE pengurangan baris kedua dan ketiga dengan baris pertama dilakukan untuk menghilangkan angka *non-zero*.

$$\left[ \begin{array}{ccc|c} 1 & 1 & -1 & -2 \\ 0 & -3 & 3 & 9 \\ 0 & 3 & 1 & -1 \end{array} \right]$$

Didapatkan elemen bernilai -3 sebagai *leading one* pada baris kedua. Untuk mendapatkan *leading one* baris ketiga, dilakukan OBE dengan menjumlahkan baris kedua ke baris ketiga.

$$\left[ \begin{array}{ccc|c} 1 & 1 & -1 & -2 \\ 0 & -3 & 3 & 9 \\ 0 & 0 & 4 & 8 \end{array} \right]$$

Baris kedua dan baris ketiga dapat disederhanakan dengan mengalikan baris dengan suatu konstanta. Misalnya baris kedua dikalikan dengan  $\frac{-1}{3}$  dan baris ketiga dikalikan dengan  $\frac{1}{4}$ .

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

Dari hasil metode Eliminasi Gauss, didapatkan :

$$x_3 = 2, x_2 - x_3 = -3, x_1 + x_2 - x_3 = -2$$

Dengan melakukan substitusi, didapatkan :

$$x_3 = 2, x_2 = -1, x_1 = 1$$

### c. Metode Eliminasi Gauss-Jordan

Prinsip yang dimiliki sama dengan Gauss, hanya saja hasil akhir dari Gauss-Jordan adalah matriks eselon baris tereduksi (*reduced row echelon form*). Pada Gauss-Jordan dilakukan *backward substitution* sedangkan pada Gauss tidak.

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

Matrix 1. Matriks hasil metode Eliminasi Gauss

$$\left[ \begin{array}{ccc|c} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & 1 & 2 \end{array} \right]$$

Matrix 2. Matriks hasil metode Eliminasi Gauss-Jordan

Langkah-langkah untuk mencari solusi dari SPL dengan metode Gauss-Jordan adalah:

1. Fase maju atau fase eliminasi gauss (sama dengan metode gauss diatas).
2. Fase mundur atau Backward Phase (menghasilkan 0 diatas leading one).

Masih menggunakan contoh yang sama dengan metode Eliminasi Gauss, dilakukan OBE sampai tahap akhir Eliminasi Gauss.

$$\left[ \begin{array}{ccc|c} 1 & 1 & -1 & -2 \\ 2 & -1 & 1 & 5 \\ -1 & 2 & 2 & 1 \end{array} \right]$$

Setelah dilakukan OBE dengan metode Eliminasi Gauss, akan didapatkan matriks seperti ini.

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

Lalu, dilakukan fase mundur dengan melakukan OBE untuk menghasilkan 0 di atas *leading one*. Baris ketiga dijumlahkan ke baris pertama dan baris kedua.

$$\left[ \begin{array}{ccc|c} 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & 1 & 2 \end{array} \right]$$

Untuk menghasilkan 0 di atas *leading one* baris kedua, baris pertama dikurangi dengan baris kedua.

$$\left[ \begin{array}{ccc|c} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & 1 & 2 \end{array} \right]$$

Maka, didapatkan hasil solusi persamaan linear dengan

$$x_3 = 2, x_2 = -1, x_1 = 1$$

#### d. Matriks Balikan (*Inverse*)

Balikan atau *inverse* dari suatu matriks akan menghasilkan matriks identitas jika dikali dengan matriks sebelum dilakukan inverse.  $A^{-1} \cdot A = I$ . Penggunaan matriks balikan ini umumnya ditujukan untuk mencari solusi dari Sistem Persamaan Linear yang memiliki solusi Tunggal.

Ada dua metode untuk mencari *invers*, yaitu metode kofaktor dan metode eliminasi Gauss-Jordan.

##### 1. Metode Kofaktor

Dalam metode kofaktor untuk mencari matriks balikan, langkah pertama adalah mencari determinan matriks tersebut. Jika suatu matriks tidak memiliki determinan, maka matriks tersebut tidak memiliki matriks balikan. Namun, jika matriks memiliki determinan, langkah berikutnya adalah mencari matriks kofaktor. Setelah itu, matriks kofaktor akan ditranspose untuk menghasilkan matriks adjoint. Di bawah ini adalah rumus untuk metode kofaktor dalam mencari matriks balikan.

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

##### 2. Metode Gauss-Jordan

*Invers* pada metode Gauss-Jordan diperoleh dengan mengubah Matriks Kiri menjadi identitas dan matriks kanan yang awalnya identitas menjadi sebuah *inverse*

$$[A|I] \rightarrow (\text{Gauss - Jordan}) \rightarrow [I|A^{-1}]$$

e. Matriks Kofaktor

Matriks kofaktor merupakan matriks yang terbentuk oleh kofaktor-kofaktor dari Matriks awalnya. Susunan elemen pada matriks ini juga sesuai dengan Matriksnya.

$$\begin{matrix} c_{11} & c_{12} & \dots & c_{1k} \\ c_{21} & c_{22} & \dots & c_{2k} \\ \dots & \dots & \dots & \dots \\ c_{k1} & c_{k2} & \dots & c_{kk} \end{matrix}$$

f. Matriks Adjoin

Adjoin adalah matriks persegi yang merupakan transpose dari matriks kofaktor, penulisannya biasanya dengan  $\text{adj}(Matriks)$ . Adjoin dari suatu matriks biasanya digunakan untuk mencari *inverse* matriks.

g. Kaidah Cramer

Kaidah Cramer digunakan untuk mencari solusi dari sistem persamaan linear yang memiliki banyak variabel dan valid saat sistem memiliki solusi Tunggal. Cara mencari solusinya adalah dengan melakukan determinant matriks koefisien dan determinant matriks lain yang diperoleh dengan mengganti salah satu kolom matriks koefisien dengan matriks hasil yang ada di sebelah kanan persamaan. Misal  $Ax = b$  merupakan SPL yang terdiri dari  $k$  persamaan linear dengan  $k$  variabel banyaknya. Jika  $\det(A) \neq 0$ , maka setiap variabel akan memiliki solusi yang unik, yaitu:

$$x_1 = \frac{\det(A_1)}{\det(A)}, \quad x_2 = \frac{\det(A_2)}{\det(A)}, \dots, \quad x_k = \frac{\det(A_k)}{\det(A)}$$

## 2. Interpolasi Polinomial

Interpolasi Polinomial adalah suatu metode analisis numerik yang melakukan aproksimasi suatu nilai fungsi polinom berdasarkan data dari titik-titik yang diketahui. Interpolasi Polinom digunakan untuk memodelkan bagaimana data berubah-ubah berdasarkan variabel independennya. Selain melakukan aproksimasi, Interpolasi Polinomial juga dapat digunakan untuk menganalisis hubungan antara variabel-variabel data.

Interpolasi polinomial dilakukan dengan cara mencari titik-titik yang akan diinterpolasi. Titik tersebut akan diperiksa nilai  $y$  dan  $x$  nya. Kemudian, akan dibentuk sebuah matriks dengan jumlah baris sebanyak  $n + 1$  titik, dan kolom sebanyak jumlah kolom + 1. Setiap kolom akan diisi nilai  $x$  dari pangkat nol sampai pangkat  $n$  dan kolom terakhir diisi dengan titik  $y$  dari titik  $x$  di baris yang sama. Untuk mendapatkan koefisien dari polinomial, dilakukan Eliminasi Gauss-Jordan pada matriks tersebut. Akan didapatkan solusi berupa persamaan polinomial berderajat  $n$  dari matriks yang telah direduksi tersebut.

### 3. Regresi Berganda

Regresi Linear Berganda adalah sebuah metode statistika yang digunakan untuk menganalisis hubungan antara beberapa variabel independen dan satu variabel dependen. Berbeda dengan Interpolasi Polinom yang dapat digunakan untuk membuat polinomial dengan derajat tertentu, Regresi Linear Berganda fokus pada pembuatan aproksimasi linear dari data yang berbeda. Meskipun demikian, Regresi Linear Berganda dapat digunakan untuk memodelkan beberapa variabel independen secara bersamaan, sedangkan Interpolasi Polinom hanya cocok untuk satu variabel independen. Kedua metode memiliki persamaan dalam hal memperkirakan hubungan antara variabel independen, seperti hubungan berbanding terbalik, berhubungan, atau berbanding lurus/linear. Persamaan umum dari regresi linear berganda adalah,

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

Kemudian, untuk mencari nilai dari setiap  $\beta_i$  dapat membuat matriks augmented dari *Normal Estimation Equation for Multiple Linear Regression* sebagai berikut

$$\left[ \begin{array}{ccccc} nb_0 & b_1 \sum_{i=1}^n x_{1i} & b_2 \sum_{i=1}^n x_{2i} & \dots & b_1 \sum_{i=1}^n x_{ki} \quad b_1 \sum_{i=1}^n y_i \\ b_0 \sum_{i=1}^n x_{1i} & b_1 \sum_{i=1}^n x_{1i}^2 & b_2 \sum_{i=1}^n x_{1i}x_{2i} & \dots & b_1 \sum_{i=1}^n x_{1i}x_{ki} \quad b_1 \sum_{i=1}^n x_{1i}y_i \\ \vdots & \vdots & \vdots & \dots & \vdots \\ b_0 \sum_{i=1}^n x_{1i} & b_1 \sum_{i=1}^n x_{ki}x_{1i} & b_2 \sum_{i=1}^n x_{ki}x_{2i} & \dots & b_1 \sum_{i=1}^n x_{ki}^2 \quad b_1 \sum_{i=1}^n x_{ki}y_i \end{array} \right]$$

Matriks tersebut kemudian direduksi dengan metode eliminasi gauss.

### 4. Bicubic Interpolation

Interpolasi bikubik adalah sebuah teknik dalam analisis numerik yang digunakan untuk mengestimasi sebuah fungsi atau sekumpulan data dengan menggunakan polinom bikubik. Polinom bikubik adalah jenis polinom dengan derajat empat yang digunakan untuk mengaproksimasi nilai-nilai di antara titik-titik data yang sudah ada. Model yang digunakan adalah sebagai berikut

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

Selain model dasar, digunakan juga model turunan berarah dari kedua sumbu. Persamaannya adalah sebagai berikut

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

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

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

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

Dari turunan berarah, dapat terbentuk sebuah matriks solusi X dengan persamaan sebagai berikut

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

Vektor  $a$  bisa didapatkan dengan mengalikan invers dari  $X$  dengan  $y$ . Vektor  $a$  tersebut kemudian dapat digunakan sebagai nilai variable dalam  $f(x, y)$ , yang membentuk fungsi interpolasi bicubic sesuai dengan model.

## BAB III

Secara garis besar, terdapat enam package pada source code. Berikut merupakan penjelasannya masing-masing:

### 1. Application

Merupakan tempat ditulisnya kode yang berhubungan dengan tiga fungsi utama pada aplikasi, yaitu Bicubic Spline, Regresi Linear Berganda, dan Interpolasi Polinomial.

### 2. CLI

Hanya pada package ini kode berinteraksi dengan dunia luar. Hal ini berarti terdapat pemisahan yang jelas antara bagian kode fungsi dan IO. Package ini memiliki dua subpackage

- a. IO, bertanggung-jawab melakan formatting dan pembacaan file
- b. Menu, sebagai user interface untuk setiap menu yang ada

### 3. Image

Package tempat seluruh *image processing* untuk melakukan scaling grayscale dilakukan. Strategi penskalaan gambar yang digunakan adalah bicubic spline interpolation.

### 4. Matrix

Tipe data matrix sendiri didefinisikan secara sederhana. Sedangkan untuk melakukan manipulasi pada matrix, diperlukan pembuatan objek baru secara eksplisit. Misalkan menggunakan MatrixManipulator. Tujuan dari pemisahan manipulasi dengan tipe data, agar kita secara tidak langsung melakukan perubahan pada matrix awal. Sehingga tidak terjadi hal yang tidak terduga ketika menggunakan tipe data matrix.

### 5. Transformation

Merupakan sebuah konsep untuk melakukan transformasi sebuah vektor untuk membentuk vektor baru. Misalkan ketika kita memiliki vektor yang berisi nilai *pixel* dari sebuah gambar, lalu kita bisa melakukan transformasi menjadi vektor yang berisi gradien yang dapat digunakan untuk melakukan aproksimasi bicubic spline.

### 6. Vector

Sebuah kumpulan dari bilangan yang memiliki basisnya masing-masing. Hal ini digeneralisasikan dengan vector space. Pada package ini memiliki dua vector space, yaitu euclidean space untuk merepresentasikan vektor dalam koordinat euclid. Berserta equation space, untuk merepresentasikan vector space yang memiliki basis seperti polinomial.

## BAB IV

Input	Output
Solusi SPL $AX=b$	
Bagian A	<pre>Matrix &gt; Linear Equations &gt; Input Masukkan banyaknya persamaan: 4 Masukkan banyaknya koefisien: 4 Masukan matrix augmented: 1 1 -1 -1 1 2 5 -7 -5 -2 2 -1 1 3 4 5 2 -4 2 6</pre> <pre>Matrix &gt; Linear Equations &gt; Result Gauss-Jordan Elimination Terdapat satu solusi Matrix setelah OBE: 1.0 0.0 0.0 0.0 0.0 0.0 1.0 0.0 0.0 2.4019198012642656E15 0.0 0.0 1.0 0.0 9.007199254740996E15 0.0 0.0 0.0 1.0 9.007199254740998E15  Bounded variable: (0.0)x<sub>0</sub> + (0.0)x<sub>1</sub> + (0.0)x<sub>2</sub> + (1.0)x<sub>3</sub> = 9.007199254740998E15 x<sub>3</sub> = 9.007199254740998E15  Bounded variable: (0.0)x<sub>0</sub> + (0.0)x<sub>1</sub> + (1.0)x<sub>2</sub> + (0.0)x<sub>3</sub> = 9.007199254740996E15 x<sub>2</sub> = 9.007199254740996E15  Bounded variable: (0.0)x<sub>0</sub> + (1.0)x<sub>1</sub> + (0.0)x<sub>2</sub> + (0.0)x<sub>3</sub> = 2.4019198012642656E16 x<sub>1</sub> = 2.4019198012642656E16  Bounded variable: (1.0)x<sub>0</sub> + (0.0)x<sub>1</sub> + (0.0)x<sub>2</sub> + (0.0)x<sub>3</sub> = -6.004799503160664E15 x<sub>0</sub> = -6.004799503160664E15  Save dalam file (Y/N)?</pre>
Bagian B	<pre>Matrix &gt; Linear Equations &gt; Input Masukkan banyaknya persamaan: 4 Masukkan banyaknya koefisien: 5 Masukan matrix augmented: 1 -1 0 0 1 3 1 1 0 -3 0 6 2 -1 0 1 -1 5 -1 2 0 -2 -1 -1</pre> <pre>Matrix &gt; Linear Equations &gt; Result Gauss-Jordan Elimination Terdapat banyak solusi Matrix setelah OBE: 1.0 0.0 0.0 0.0 3.0 0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 1.0 0.0 -1.0 0.0 0.0 0.0 0.0 1.0 0.0  Bounded variable: (0.0)x<sub>0</sub> + (0.0)x<sub>1</sub> + (0.0)x<sub>2</sub> + (0.0)x<sub>3</sub> + (1.0)x<sub>4</sub> = 0.0 x<sub>4</sub> = 0.0  Bounded variable: (0.0)x<sub>0</sub> + (0.0)x<sub>1</sub> + (0.0)x<sub>2</sub> + (1.0)x<sub>3</sub> + (0.0)x<sub>4</sub> = -1.0 x<sub>3</sub> = -1.0  Free variable: x<sub>2</sub> = t<sub>1</sub>  Bounded variable: (0.0)x<sub>0</sub> + (1.0)x<sub>1</sub> + (0.0)x<sub>2</sub> + (0.0)x<sub>3</sub> + (0.0)x<sub>4</sub> = 0.0 x<sub>1</sub> = 0.0  Bounded variable: (1.0)x<sub>0</sub> + (0.0)x<sub>1</sub> + (0.0)x<sub>2</sub> + (0.0)x<sub>3</sub> + (0.0)x<sub>4</sub> = 3.0 x<sub>0</sub> = 3.0  Save dalam file (Y/N)?</pre>
Bagian C	<pre>Matrix &gt; Linear Equations &gt; Input Masukkan banyaknya persamaan: 3 Masukkan banyaknya koefisien: 6 Masukan matrix augmented: 0 1 0 0 1 0 2 0 0 0 1 1 0 -1 0 1 0 0 0 1 1</pre> <pre>Matrix &gt; Linear Equations &gt; Result Gauss-Jordan Elimination Terdapat banyak solusi Matrix setelah OBE: 0.0 1.0 0.0 0.0 0.0 1.0 -1.0 0.0 0.0 0.0 1.0 0.0 1.0 -2.0 0.0 0.0 0.0 0.0 1.0 -1.0 1.0  Free variable: x<sub>5</sub> = t<sub>1</sub>  Bounded variable: (0.0)x<sub>0</sub> + (0.0)x<sub>1</sub> + (0.0)x<sub>2</sub> + (0.0)x<sub>3</sub> + (1.0)x<sub>4</sub> + (-1.0)x<sub>5</sub> = 1.0 x<sub>4</sub> = (1.0)t<sub>1</sub> + 1.0  Bounded variable: (0.0)x<sub>0</sub> + (0.0)x<sub>1</sub> + (0.0)x<sub>2</sub> + (1.0)x<sub>3</sub> + (0.0)x<sub>4</sub> + (1.0)x<sub>5</sub> = -2.0 x<sub>3</sub> = (-1.0)t<sub>1</sub> + -2.0  Free variable: x<sub>2</sub> = t<sub>2</sub>  Bounded variable: (0.0)x<sub>0</sub> + (1.0)x<sub>1</sub> + (0.0)x<sub>2</sub> + (0.0)x<sub>3</sub> + (0.0)x<sub>4</sub> + (1.0)x<sub>5</sub> = 1.0 x<sub>1</sub> = (-1.0)t<sub>1</sub> + 1.0  Free variable: x<sub>0</sub> = t<sub>3</sub>  Save dalam file (Y/N)?</pre>
Bagian D Matrix Hilbert	

Matrix > Linear Equations > Input

Masukkan banyaknya persamaan: 6

Masukkan banyaknya koefisien: 6

Masukan matrix augmented:

```
1 0.5 0.33333 0.25 0.2 0.16666 1
0.5 0.33333 0.25 0.2 0.16666 0.14285 0
0.33333 0.25 0.2 0.16666 0.14285 0.125 0
0.25 0.2 0.16666 0.14285 0.125 0.11111 0
0.2 0.16666 0.14285 0.125 0.11111 0.1 0
0.16666 0.14285 0.125 0.11111 0.1 0.090909 0
```

```
Free variable:
x0 = t1

Bounded variable:
(0.0)x0 + (0.0)x1 + (0.0)x2 + (1.0)x3 + (1.0)x4 + (1.0)x5 = 92.18886412989611
x4 = (-1.8165789772440617)t1 + 92.18886412989611

Bounded variable:
(0.0)x0 + (0.0)x1 + (0.0)x2 + (1.0)x3 + (0.0)x4 + (1.0011420982914205E15)x5 = 2.96354314262242E17
x5 = (-1.0011420982914205E15)t1 + 2.96354314262242E17

Bounded variable:
(0.0)x0 + (0.0)x1 + (4.0)x2 + (0.0)x3 + (0.0)x4 + (-1.670282192742145E30)x5 = -4.944306454598978E32
x5 = (1.6721896191806686E45)t1 + -4.94995333855084E47

Bounded variable:
(0.0)x0 + (1.0)x1 + (0.0)x2 + (0.0)x3 + (7.7947496154483E29)x4 = 2.307292471324522E32
x4 = (-7.00338157766567E44)t1 + 2.309927626113833E47

Bounded variable:
(1.0)x0 + (0.0)x1 + (0.0)x2 + (0.0)x3 + (-0.854018674382268E28)x4 = -2.472925151989475E31
x4 = (3.635597845367963)t1 + -2.475749475588377E46

Save dalam file (Y/N)?
```

Matrix > Linear Equations > Input

Masukkan banyaknya persamaan: 10

Masukkan banyaknya koefisien: 10

Masukan matrix augmented:

```
1 0.5 0.33333333 0.25 0.2 0.1666667 0.14285714 0.125 0.11111111 0.1 1
0.5 0.33333333 0.25 0.2 0.16666667 0.14285714 0.125 0.11111111 0.1 0.09090909 0.08333333 0
0.33333333 0.25 0.2 0.16666667 0.14285714 0.125 0.11111111 0.1 0.09090909 0.08333333 0.07692308 0
0.25 0.2 0.16666667 0.14285714 0.125 0.11111111 0.1 0.09090909 0.08333333 0.07692308 0.07142857 0
0.16666667 0.14285714 0.125 0.11111111 0.1 0.09090909 0.08333333 0.07692308 0.07142857 0.0625 0.05882353 0
0.14285714 0.125 0.11111111 0.1 0.09090909 0.08333333 0.07692308 0.07142857 0.0625 0.05882353 0.0563158 0
0.125 0.11111111 0.1 0.09090909 0.08333333 0.07692308 0.07142857 0.0625 0.05882353 0.0563158 0
0.1 0.09090909 0.08333333 0.07692308 0.07142857 0.0625 0.05882353 0.0563158 0
0.1 0.09090909 0.08333333 0.07692308 0.07142857 0.0625 0.05882353 0.0563158 0
0.1 0.09090909 0.08333333 0.07692308 0.07142857 0.0625 0.05882353 0.0563158 0
0.1 0.09090909 0.08333333 0.07692308 0.07142857 0.0625 0.05882353 0.0563158 0
```

```
Tersedia satu solusi
1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
x0 = 387.72338855875
x1 = 0.0
x2 = 0.0
x3 = 0.0
x4 = 0.0
x5 = 0.0
x6 = 0.0
x7 = 0.0
x8 = 0.0
x9 = 0.0
x10 = 0.0

Bounded variable:
(0.0)x0 + (0.0)x1 + (0.0)x2 + (0.0)x3 + (0.0)x4 + (0.0)x5 + (0.0)x6 + (0.0)x7 + (0.0)x8 + (0.0)x9 + (0.0)x10 = -13766.8127799238
x10 = -13766.8127799238

Bounded variable:
(0.0)x0 + (0.0)x1 + (0.0)x2 + (0.0)x3 + (0.0)x4 + (0.0)x5 + (0.0)x6 + (0.0)x7 + (0.0)x8 + (0.0)x9 + (0.0)x10 = -2667.39811708986
x10 = -2667.39811708986

Bounded variable:
(0.0)x0 + (0.0)x1 + (0.0)x2 + (0.0)x3 + (0.0)x4 + (0.0)x5 + (0.0)x6 + (0.0)x7 + (0.0)x8 + (0.0)x9 + (0.0)x10 = -4728.45985450808
x10 = -4728.45985450808

Bounded variable:
(0.0)x0 + (0.0)x1 + (0.0)x2 + (0.0)x3 + (0.0)x4 + (0.0)x5 + (0.0)x6 + (0.0)x7 + (0.0)x8 + (0.0)x9 + (0.0)x10 = -2783.30380006413
x10 = -2783.30380006413

Bounded variable:
(0.0)x0 + (0.0)x1 + (0.0)x2 + (0.0)x3 + (0.0)x4 + (0.0)x5 + (0.0)x6 + (0.0)x7 + (0.0)x8 + (0.0)x9 + (0.0)x10 = -7216.9560837951
x10 = -7216.9560837951

Bounded variable:
(0.0)x0 + (0.0)x1 + (0.0)x2 + (0.0)x3 + (0.0)x4 + (0.0)x5 + (0.0)x6 + (0.0)x7 + (0.0)x8 + (0.0)x9 + (0.0)x10 = -24796.94986479627
x10 = -24796.94986479627

Bounded variable:
(0.0)x0 + (0.0)x1 + (0.0)x2 + (0.0)x3 + (0.0)x4 + (0.0)x5 + (0.0)x6 + (0.0)x7 + (0.0)x8 + (0.0)x9 + (0.0)x10 = -22564.1598000792
x10 = -22564.1598000792

Bounded variable:
(0.0)x0 + (0.0)x1 + (0.0)x2 + (0.0)x3 + (0.0)x4 + (0.0)x5 + (0.0)x6 + (0.0)x7 + (0.0)x8 + (0.0)x9 + (0.0)x10 = -7055.679532780774
x10 = -7055.679532780774

Bounded variable:
(0.0)x0 + (0.0)x1 + (0.0)x2 + (0.0)x3 + (0.0)x4 + (0.0)x5 + (0.0)x6 + (0.0)x7 + (0.0)x8 + (0.0)x9 + (0.0)x10 = -3873.72338855875
x10 = -3873.72338855875

Bounded variable:
(0.0)x0 + (0.0)x1 + (0.0)x2 + (0.0)x3 + (0.0)x4 + (0.0)x5 + (0.0)x6 + (0.0)x7 + (0.0)x8 + (0.0)x9 + (0.0)x10 = -46.41389328936082
x10 = -46.41389328936082
```

## SPL Matriks Augmented

### Bagian A

Matrix > Linear Equations > Input

Masukkan banyaknya persamaan: 4

Masukkan banyaknya koefisien: 4

Masukan matrix augmented:

```
1 -1 2 -1
2 1 -2 -2
-1 2 4 1
3 0 0 -3 -3
```

Matrix > Linear Equations > Result Gauss-Jordan Elimination

Terdapat satu solusi

Matrix setelah OBE:

```
1.0 0.0 0.0 1.0 1.0
0.0 1.0 2.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0
```

Free variable:

$x_3 = t_1$

Free variable:

$x_4 = t_2$

Bounded variable:

```
(0.0)x0 + (1.0)x1 + (2.0)x2 + (0.0)x3 = 0.0
x4 = (-2.0)t2
```

Bounded variable:

```
(1.0)x0 + (0.0)x1 + (0.0)x2 + (1.0)x3 = 1.0
x0 = 1.0 + (-1.0)t1
```

### Bagian B

Matrix > Linear Equations > Input

Masukkan banyaknya persamaan: 6

Masukkan banyaknya koefisien: 4

Masukan matrix augmented:

```
2 0 8 0
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
```

Matrix > Linear Equations > Result Gaussian Elimination

Terdapat satu solusi

Matrix setelah OBE:

```
1.0 0.0 -1.5 0.0 -1.5
0.0 1.0 0.0 -1.5 0.5
0.0 0.0 1.0 0.0 1.0
0.0 0.0 0.0 1.0 1.0
0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0
```

Bounded variable:

```
(1.0)x0 + (0.0)x1 + (0.0)x2 + (1.0)x3 = 1.0
x3 = 1.0
```

Bounded variable:

```
(0.0)x0 + (0.0)x1 + (1.0)x2 + (0.0)x3 = 1.0
x2 = 1.0
```

Bounded variable:

```
(0.0)x0 + (1.0)x1 + (0.0)x2 + (-1.5)x3 = 0.5
x1 = 2.0
```

Bounded variable:

```
(1.0)x0 + (0.0)x1 + (-1.5)x2 + (0.0)x3 = -1.5
x0 = 0.0
```

Save dalam file (Y/N)?

## SPL Lainnya

### Bagian A

<pre>Matrix &gt; Linear Equations &gt; Input Masukkan banyaknya persamaan: 4 Masukkan banyaknya koefisien: 4 Masukan matrix augmented: 8 1 3 2 0 2 9 -1 -2 1 1 3 2 -1 2 1 0 6 4 3</pre>	<pre>Matrix &gt; Linear Equations &gt; Result Gauss-Jordan Elimination Terdapat satu solusi Matrix setelah OBE: 1.0 0.0 0.0 0.0 -0.2243243243243243 0.0 1.0 0.0 0.0 0.18243243243243243 0.0 0.0 1.0 0.0 0.7094594594594594 0.0 0.0 0.0 1.0 -0.2581081081081081  Bounded variable: (0.0)x<sub>0</sub> + (0.0)x<sub>1</sub> + (1.0)x<sub>2</sub> + (1.0)x<sub>3</sub> = -0.2581081081081081 x<sub>3</sub> = -0.2581081081081081  Bounded variable: (0.0)x<sub>0</sub> + (0.0)x<sub>1</sub> + (1.0)x<sub>2</sub> + (0.0)x<sub>3</sub> = 0.7094594594594594 x<sub>2</sub> = 0.7094594594594594  Bounded variable: (0.0)x<sub>0</sub> + (0.0)x<sub>1</sub> + (0.0)x<sub>2</sub> + (0.0)x<sub>3</sub> = 0.1824324324324324 x<sub>1</sub> = 0.1824324324324324  Bounded variable: (1.0)x<sub>0</sub> + (0.0)x<sub>1</sub> + (0.0)x<sub>2</sub> + (0.0)x<sub>3</sub> = -0.2243243243243243 x<sub>0</sub> = -0.2243243243243243  Save dalam file (Y/N)?</pre>
---	--

## Bagian B

<pre>Matrix &gt; Linear Equations &gt; Input Masukkan banyaknya persamaan: 12 Masukkan banyaknya koefisien: 9 Masukan matrix augmented: 0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 1.00000 1.00000 1.00000 13.0000 0.00000 0.00000 0.00000 1.00000 0.00000 0.00000 0.00000 0.00000 0.00000 15.0000 1.00000 1.00000 0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 8.00000 0.00000 0.00000 0.04289 0.00000 0.04289 0.75000 0.04289 0.75000 0.61396 14.7900 0.00000 0.25000 0.91421 0.25000 0.91421 0.25000 0.91421 0.25000 0.00000 14.3100 0.61396 0.75000 0.04289 0.75000 0.04289 0.00000 0.04289 0.00000 3.81000 0.00000 0.00000 1.00000 0.00000 1.00000 0.00000 1.00000 0.00000 1.00000 18.0000 0.00000 1.00000 0.00000 0.00000 1.00000 0.00000 0.00000 1.00000 0.00000 12.0000 1.00000 0.00000 1.00000 0.00000 1.00000 0.00000 0.00000 1.00000 0.00000 6.00000 0.04289 0.75000 0.61396 0.00000 0.04289 0.75000 0.00000 0.04289 10.5100 0.91421 0.25000 0.00000 0.25000 0.91421 0.25000 0.00000 0.25000 0.91421 16.1300 0.04289 0.00000 0.75000 0.04289 0.00000 0.61396 0.75000 0.04289 7.04000</pre>	<pre>Matrix &gt; Linear Equations &gt; Result Gauss-Jordan Elimination Tidak memiliki solusi Save dalam file (Y/N)?</pre>
---	---

## Sistem Persamaan Berdasarkan Reaktor

<pre>Matrix &gt; Linear Equations &gt; Input Masukkan banyaknya persamaan: 3 Masukkan banyaknya koefisien: 3 Masukan matrix augmented: -120 60 0 -1300 40 -80 0 0 80 20 -150 -200</pre>	<pre>Matrix &gt; Linear Equations &gt; Result Gauss-Jordan Elimination Terdapat satu solusi Matrix setelah OBE: 1.0 0.0 0.0 14.44444444444445 0.0 1.0 0.0 7.222222222222222 0.0 0.0 1.0 10.0  Bounded variable: (0.0)x<sub>0</sub> + (0.0)x<sub>1</sub> + (1.0)x<sub>2</sub> = 10.0 x<sub>2</sub> = 10.0  Bounded variable: (0.0)x<sub>0</sub> + (1.0)x<sub>1</sub> + (0.0)x<sub>2</sub> = 7.222222222222222 x<sub>1</sub> = 7.222222222222222  Bounded variable: (1.0)x<sub>0</sub> + (0.0)x<sub>1</sub> + (0.0)x<sub>2</sub> = 14.44444444444445 x<sub>0</sub> = 14.44444444444445  Save dalam file (Y/N)?</pre>
---	--

## Studi Kasus Interpolasi

### Interpolasi Berdasarkan Tabel

<pre>Matrix &gt; Polynomial Interpolation &gt; Input Masukkan banyaknya sampel: 7 Masukkan sampel: 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 Masukkan x untuk interpolasi:</pre>	<pre>Matrix &gt; Polynomial Interpolation &gt; Result -0.02297656250000037 + 0.24000000000000643x<sup>1</sup> + 0.1973958333329772x<sup>2</sup> + 8.96782648149702E-14x<sup>3</sup> + 0.0260416666665538x<sup>4</sup> + 6.907553684857869E-14x<sup>5</sup> + -1.6340677274111618E-14x<sup>6</sup> f(0.2) = 0.0329609375000006 Save dalam file (Y/N)?</pre>
	<pre>Matrix &gt; Polynomial Interpolation &gt; Result -0.02297656250000037 + 0.24000000000000643x<sup>1</sup> + 0.1973958333329772x<sup>2</sup> + 8.96782648149702E-14x<sup>3</sup> + 0.0260416666665538x<sup>4</sup> + 6.907553684857869E-14x<sup>5</sup> + -1.6340677274111618E-14x<sup>6</sup> f(0.55) = 0.17111865234375007 Save dalam file (Y/N)?</pre>
	<pre>Matrix &gt; Polynomial Interpolation &gt; Result -0.02297656250000037 + 0.24000000000000643x<sup>1</sup> + 0.1973958333329772x<sup>2</sup> + 8.96782648149702E-14x<sup>3</sup> + 0.0260416666665538x<sup>4</sup> + 6.907553684857869E-14x<sup>5</sup> + -1.6340677274111618E-14x<sup>6</sup> f(0.85) = 0.33723583984374994 Save dalam file (Y/N)?</pre>

	Matrix > Polynomial Interpolation > Result $-0.02297656250000037 + 0.2400000000000643x^1 + 0.1973958333329772x^2 + 8.967826481409702E-14x^3 + 0.0260416666665538x^4 + 6.907553684857869E-14x^5 + -1.6340677274111618E-14x^6$ $f(1.28) = 0.6775418375000001$ Save dalam file (Y/N)? ■
--	---

## Interpolasi Jumlah Kasus Baru Covid-19

Matrix > Polynomial Interpolation > Input Masukkan banyaknya sampel: 10 Masukkan sampel: 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 Masukkan x untuk interpolasi: ■	Matrix > Polynomial Interpolation > Result $7.187700143017766E12 + -0.34773205458011E12x^1 + 5.334585430409406E12x^2 + -1.7569254832327988E12x^3 + 3.685731332143059E11x^4 + -5.113475592422699E10x^5 + 4.696053591059929E9x^6 + -2.7548817775286216E8x^7 + 9373287.57531152x^8 + -140999.96712447252x^9$ $f(7.516) = 53537.90234375$ Save dalam file (Y/N)? ■
	Matrix > Polynomial Interpolation > Result $7.187700143017766E12 + -0.34773205458011E12x^1 + 5.334585430409406E12x^2 + -1.7569254832327988E12x^3 + 3.685731332143059E11x^4 + -5.113475592422699E10x^5 + 4.696053591059929E9x^6 + -2.7548817775286216E8x^7 + 9373287.57531152x^8 + -140999.96712447252x^9$ $f(9.166) = -659026.9765625$ Save dalam file (Y/N)? ■

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

Matrix > Polynomial Interpolation > Input Masukkan banyaknya sampel: 5 Masukkan sampel: 0.4 0.418 0.8 0.507 1.2 0.560 1.6 0.583 2 0.576 Masukkan x untuk interpolasi: ■	Matrix > Polynomial Interpolation > Result $0.281000000000006 + 0.416249999999968x^1 + -0.204687499999946x^2 + 0.0546874999999647x^3 + -0.00976562499999226x^4$ $f(1.0) = 0.537484375$ Save dalam file (Y/N)? ■
---	--

## Studi Kasus Regresi Linear Berganda

Matrix > Multiple Linear Regression > Input Masukkan banyak peubah: 3 Masukkan banyak sampel: 20 Masukkan sample point 72.4 76.3 29.18 0.90 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 1.00 12.9 67.4 29.39 1.10 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 29.60 1.07 23.2 76.8 29.38 1.07 Matrix > Multiple Linear Regression > Result $20.0\ 863.099999999999\ 1530.400000000003\ 587.839999999999\ 19.42$ $863.099999999999\ 54876.89\ 67000.09\ 25283.395\ 779.476999999999$ $1530.400000000003\ 67000.09\ 117912.3200000002\ 44976.86699999984\ 1483.436999999999$ $587.839999999999\ 25283.395\ 44976.86699999984\ 17278.508600000005\ 571.121900000001$ $x_0 = 50.0$ $x_1 = 76.0$ $x_2 = 29.3$ $-3.507778140862092 + (-0.002624990745881746)x_1 + (7.989410472207176E-4)x_2 + (.0.15415503019758064)x_3$ Hasil regresi: 0.9384342262217085	
---	--

## Studi Kasus Interpolasi Bicubic Spline

```

Matrix > Bicubic Spline > Input
f(0,0) = 21
f(1,0) = 98
f(0,1) = 125
f(1,1) = 153
f_x(0,0) = 51
f_x(1,0) = 101
f_x(0,1) = 161
f_x(1,1) = 59
f_y(0,0) = 0
f_y(1,0) = 42
f_y(0,1) = 72
f_y(1,1) = 210
f_xy(0,0) = 16
f_xy(1,0) = 12
f_xy(0,1) = 81
f_xy(1,1) = 96
Masukkan titik: █

```

```

Matrix > Bicubic Spline > Result
(21.0) + (51.0)x^1 + (28.0)x^2 + (-2.0)x^3 + (1
6.0)x^4y^1 + (82.0)x^2y^1 + (-56.0)x^3y^1 + (240.0
)y^2 + (217.0)x^4y^2 + (-1295.0)x^2y^2 + (709.0)x
^3y^2 + (-136.0)y^3 + (-123.0)x^4y^3 + (888.0)x^2y
^3 + (-487.0)x^3y^3
Hasil aproksimasi f(0.0,0.0) = 21.0
Save dalam file (Y/N)? █

```

```

Matrix > Bicubic Spline > Result
(21.0) + (51.0)x^1 + (28.0)x^2 + (-2.0)x^3 + (1
6.0)x^4y^1 + (82.0)x^2y^1 + (-56.0)x^3y^1 + (240.0
)y^2 + (217.0)x^4y^2 + (-1295.0)x^2y^2 + (709.0)x
^3y^2 + (-136.0)y^3 + (-123.0)x^4y^3 + (888.0)x^2y
^3 + (-487.0)x^3y^3
Hasil aproksimasi f(0.5,0.5) = 87.796875
Save dalam file (Y/N)? █

```

```

Matrix > Bicubic Spline > Result
(21.0) + (51.0)x^1 + (28.0)x^2 + (-2.0)x^3 + (1
6.0)x^4y^1 + (82.0)x^2y^1 + (-56.0)x^3y^1 + (240.0
)y^2 + (217.0)x^4y^2 + (-1295.0)x^2y^2 + (709.0)x
^3y^2 + (-136.0)y^3 + (-123.0)x^4y^3 + (888.0)x^2y
^3 + (-487.0)x^3y^3
Hasil aproksimasi f(0.25,0.75) = 117.73217773
4375
Save dalam file (Y/N)? █

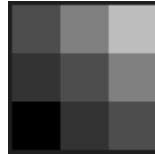
```

```

Matrix > Bicubic Spline > Result
(21.0) + (51.0)x^1 + (28.0)x^2 + (-2.0)x^3 + (1
6.0)x^4y^1 + (82.0)x^2y^1 + (-56.0)x^3y^1 + (240.0
)y^2 + (217.0)x^4y^2 + (-1295.0)x^2y^2 + (709.0)x
^3y^2 + (-136.0)y^3 + (-123.0)x^4y^3 + (888.0)x^2y
^3 + (-487.0)x^3y^3
Hasil aproksimasi f(0.1,0.9) = 128.5751870000
0003
Save dalam file (Y/N)? █

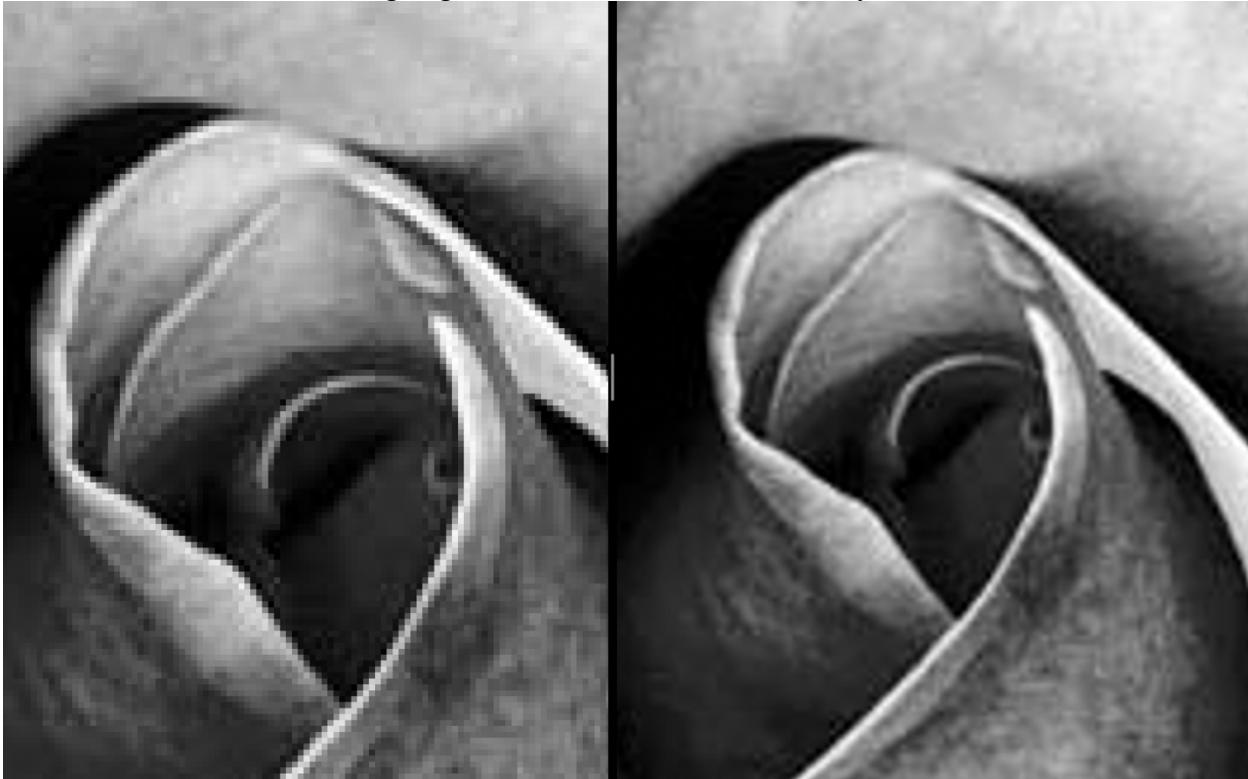
```

### Bonus : Resizing Image





Perbandingan gambar setelah diskalakan sebanyak 15 kali



## BAB V

1. Simpulan
  - a. Sudah dilakukan implementasi Library pada program seperti yang dijelaskan pada bab III
  - b. Program telah diimplementasikan untuk menyelesaikan persoalan-persoalan mengenai sistem persamaan linear, interpolasi polynominal, interpolasi bikubik, dan regresi linear seperti yang dipaparkan pada bab IV
2. Saran
  - a. Kami perlu lebih teliti dalam penggerjaan kode. Kami sering berasumsi bahwa kode sudah aman hanya untuk testcase yang sederhana.
  - b. Kami perlu lebih banyak melakukan testing dalam kode. Banyak kode yang bergantung pada kode lainnya dan kami belum melakukan testing dengan kasus-kasus yang lebih bervariasi sehingga berdampak banyak pada kode lainnya.
3. Refleksi

Kami belajar banyak dalam tugas ini. Pengalaman kami selama mengerjakan tubes ini cukup memuaskan, tetapi karena ditambahkan tekanan matkul lain dan kurang mahirnya beberapa dari kami dalam menggunakan Bahasa java sehingga ada beberapa proses yang perlu waktu lebih lama untuk diselesaikan. Kerja sama diantara kami cukup bagus, komunikasi terjaga satu sama lain dan bila ada kesulitan saling membantu satu sama lain.

## LAMPIRAN

1. Package Matrix

a. Class Matrix

i. Attribute

Nama	Tipe	Deskripsi
Row	Int	Banyak baris untuk sebuah matriks
Col	Int	Banyak Kolom untuk sebuah matriks
Data	Double[][]	Array 2 dimensi kosong untuk menampung elemen-elemen dari matriks nantinya.

ii. Method

Nama	Tipe	Parameter	Deskripsi
Matrix	Public konstruktor	Int row, int col	Konstruktor untuk membuat matrix
Get	Public double	Int row, int col	Mengambil nilai nilai elemen sebuah matrix
set	Public void	Int row, int col, int newValue	Mengganti elemen sebuah matrix
Copy	Public Matrix	-	Mereplika sebuah matrix dengan baris dan kolom serta elemen yang sama
createIdentityMatrix	Public static Matrix	Int size	Membuat matriks identitas dengan ukuran size x size.

b. Class MatrixArithmetic

i. Attribute

Nama	Tipe	Deskripsi
M1, M2	Matrix	Sebuah Matriks

ii. Method

Nama	Tipe	Parameter	Deskripsi
MatrixArithmetic	Public konstruktor	Matrix M1, Matrix M2	Konstrukstor untuk class MatrixArithmetic
Addition	Public static Matrix	Matrix M1, Matrix M2	Melakukan penjumlahan matriks
MultiplyByConst	Public static Matrix	Matrix M, double constant	Melakukan perkalian tiap elemen matriks dengan constant

Subtraction	Public static Matrix	Matrix M1, Matrix M2	Melakukan pengurangan terhadap Matrix M1 dan Matrix M2
Multiply	Public static Matrix	Matrix M1, Matrix M2	Melakukan perkalian matrix M1 dengan M2

c. Class MatrixCramer

i. Attribute

Nama	Tipe	Deskripsi
-	-	-

ii. Method

Nama	Tipe	Parameter	Deskripsi
calculateSolution	Public static double[][]	Matrix M, Matrix B	Menghitung hasil solusi persamaan linear menggunakan kaidah cramer dengan parameter matriks koefisien dan konstanta
calculateAugmented	Public static double[][]	Matrix M	Menghitung hasil solusi persamaan linear dengan kaidah cramer dengan parameter matriks augmented

d. Class MatrixDeterminant

i. Attribute

Nama	Tipe	Deskripsi
-	-	-

ii. Method

Nama	Tipe	Parameter	Deskripsi
cofactor	Public static Matrix	Matrix M, int Row, int Col	Mereturn kofaktor dari sebuah elemen matriks
calculate	Public static Double	Matrix M	Menghitung determinan dari Matrix M dengan metode ekspansi kofaktor

e. Class MatrixDeterminantWithOBE

i. Attribute

Nama	Tipe	Deskripsi
-	-	-

ii. Method

Nama	Tipe	Parameter	Deskripsi
calculate	Public static double	Matrix A	Menghitung determinan dari Matrix A dengan metode eliminasi gauss

f. Class MatrixDimensionManipulator

i. Attribute

Nama	Tipe	Deskripsi
M	Matrix	Sebuah matriks

ii. Method

Nama	Tipe	Parameter	Deskripsi
MatrixDimensionManipulator	Public konstruktor	Matrix M	Konstruktor untuk class MatrixDimensionManipulator
transpose	Public void		Melakukan transpose terhadap matriks
addRowToBelow	Public void	double[] row	Menambahkan baris baru pada ujung bawah matriks dan mengisi baris tersebut dengan parameter double[] row
addColumnToRight	Public void	double[] col	Menambahkan kolom baru pada ujung kanan matriks dan mengisi kolom tersebut dengan parameter double[] col
getResult	Public Matrix	-	Mengembalikan matriks hasil operasi dari class MatrixDimensionManipulator

g. Class MatrixInverse

i. Attribute

Nama	Tipe	Deskripsi
-	-	-

ii. Method

Nama	Tipe	Parameter	Deskripsi
calculateWithCofactor	Public static Matrix	Matrix M	Menghitung invers matriks dengan adjoint dan kofaktor
adjoin	Public static Matrix	Matrix M	Menghitung adjoint dari setiap elemen pada matriks
addIdentityMatrixToRight	Private static Matrix	Matrix M	Menambahkan matriks identitas pada ujung kanan matriks

getRightSideMatrix	Private static Matrix	Matrix M	Mengembalikan seluruh elemen dari sisi kanan matriks dalam bentuk matriks
calculateWithGaussJordan	Public static Matrix	Matrix M	Menghitung invers matriks dengan metode Eliminasi Gauss Jordan

h. MatrixLinearEquation

i. Attribute

Nama	Tipe	Deskripsi
-	-	-

ii. Method

Nama	Tipe	Parameter	Deskripsi
augmentedMatrix	Private static Matrix	Matrix MatriksKoefisien, Matrix MatriksKonstanta	Mengembalikan matriks augmented dari MatriksKoefisien dan MatriksKonstanta
solution	Public static String	Matrix augmentedmatrix, int type	Mengembalikan hasil solusi persamaan linear dengan pilihan metode Eliminasi Gauss, Eliminasi Gauss Jordan, dan Kaidah Cramer.
solutionAugmented	Public static String	Matrix koefisien, Matrix konstanta, int type	Memisahkan matriks augmented agar parameter bisa digunakan di method solution
typeOfSolution	Private static int	Matrix M1	Menentukan apakah persamaan linear memiliki solusi tunggal, banyak solusi atau tidak memiliki solusi
parametricSolution	Private static String	Matrix M1	Mengembalikan string solusi untuk solusi parametrik
gaussianEliminationSolution	Private static String	Matrix M1	Mengembalikan string solusi untuk metode Eliminasi Gauss

gaussJordanEliminationSolution	Private static String	Matrix M1	Mengembalikan string solusi untuk metode Eliminasi Gauss Jordan
cramerSolution	Private static String	Matrix koefisien, Matrix konstanta	Mengembalikan string solusi untuk metode Kaidah Cramer

i. MatrixManipulator

i. Attribute

Nama	Tipe	Deskripsi
M	Matrix	Sebuah matriks
col	int	Banyak kolom dari matriks
row	int	Banyak baris dari matriks

ii. Method

Nama	Tipe	Parameter	Deskripsi
MatrixManipulator	Public konstruktor	Matrix M	Konstruktor untuk class MatrixManipulator
get	Public double	int row, int col	Mengembalikan nilai dari elemen matriks dengan indeks (row,col)
set	Public void	int row, int col, double newValue	Mengubah elemen matriks dengan indeks (row,col) menjadi newValue
getRow	Public double[]	int row	Mengembalikan array berisi elemen Matriks dengan indeks baris parameter row.
setRow	Public void	int row, double[] rows	Mengubah seluruh elemen yang ada pada baris dengan indeks parameter row di Matrix menjadi setiap elemen dari parameter rows
getCol	Public double[]	int col	Mengembalikan array berisi elemen Matriks dengan indeks kolom parameter col.
setCol	Public void	int col, double[] cols	Mengubah seluruh elemen yang ada pada kolom dengan indeks parameter col di Matrix menjadi setiap elemen dari parameter cols
getResult	Public Matrix	-	Mengembalikan matriks hasil operasi class MatrixManipulator

j. MatrixReader

i. Attribute

Nama	Tipe	Deskripsi
cliScanner	Static Scanner	Sebagai pemindai input dari terminal
file	Static BufferedReader	

ii. Method

Nama	Tipe	Parameter	Deskripsi
read	Public static Matrix	Scanner s, int row, int col	Mengembalikan matriks dengan setiap elemen dari input terminal. Jika hanya diberikan argumen Scanner s, maka akan diminta banyak baris dan kolom pada input terminal. Jika dioverload dengan int row dan int col, method akan langsung meminta elemen dari matriks
readCLI	Public static Matrix	int row, int col	Menjalankan method read. Jika tidak diberikan argumen, akan diminta input untuk banyak baris dan kolom. Jika dioverload dengan int row dan int col, maka method read akan langsung dijalankan dan mengembalikan matriks hasil method read
readFile	Public static Matrix	String fileName, int row, int col	Mengembalikan matriks hasil dari pembacaan file dengan nama file dari parameter fileName. Jika hanya diberikan argumen fileName, maka baris dan kolom menyesuaikan dengan matriks yang ada pada file. Jika dioverload dengan int row dan int col, maka matriks yang dihasilkan akan dibatasi sesuai

			dengan parameter overload tersebut.
fileRow	Public static int	String fileName	Mengembalikan jumlah baris yang ada pada matriks di dalam file dengan nama file dari parameter fileName
fileCol	Public static int	String fileName	Mengembalikan jumlah kolom yang ada pada matriks di dalam file dengan nama file dari parameter fileName
readLastLine	Public static double[]	String fileName	Mengembalikan array yang berisi elemen pada baris terakhir dari file dengan nama file dari parameter fileName
readFileCLI	Public static Matrix	int row, int col	Meminta input dari pengguna untuk file yang akan dibaca menjadi matrix. Jika tidak diberikan argumen, maka matriks yang dihasilkan akan menyesuaikan dengan matriks pada file. Jika diberikan argumen row dan col, maka baris dan kolom dari matriks yang dihasilkan akan dibatasi sesuai argumen row dan col.

2. Package Application

a. Class BicubicSpline

i. Attribute

Nama	Tipe	Deskripsi
independentVariableCount	Private static int	-
transformation	Private static Transformation	Variable untuk menampung bicubicSpline yang sudah ditransformasi.

ii. Method

Nama	Tipe	Parameter	Deskripsi

getEquation	Public static EquationSpace	EuclideanSpace v	Mengambil nilai dari Persamaan yang ada
approximate	Public static double	EquationSpace eq, double x, double y	Melakukan aproksimasi dari persamaan.

b. Class MultipleRegression

i. Attribute

Nama	Tipe	Deskripsi
-	-	-

ii. Method

Nama	Tipe	Parameter	Deskripsi
createMatrix	Private static Matrix	Matrix samplePoints	Konstruktor untuk membuat input menjadi dalam bentuk matriks
solve	Public static EqationSpace	Matrix samplePoint	Digunakan untuk menghitung hasil dari MultipleLinear dari input yang diberikan
DisplayEquation	Public static void	EquationSpace result	Menampilkan persamaan yang sudah dihitung dari input yang ada ke layar

c. Class PolynomialInterpolation

i. Attribute

Nama	Tipe	Deskripsi
-	-	-

ii. Method

Nama	Tipe	Parameter	Deskripsi
pointInput	Private static Matrix	Matrix points	Mengubah input menjadi matrix yang dapat digunakan dalam interpolasi
calculate	Public static Matrix	Matrix points	Mengembalikan matrix hasil interpolasi polinomial.
f	Public static double	Matrix points, double x	Mengembalikan nilai fungsi pada titik x hasil interpolasi

3. Package Vector

a. Class BicubicSplineSpace

i. Attribute

Nama	Tipe	Deskripsi
-	-	-

independentVariableCount	Public static int	-
maxDegree	Public static int	-

ii. Method

Nama	Tipe	Parameter	Deskripsi
BicubicSplineSpace	konstruktor	-	Konstruktor untuk bicubicsplinespace

b. Class EquationSpace

i. Attribute

Nama	Tipe	Deskripsi
independentVariableCount	public int	Variable untuk menentukan banyaknya variable dalam equationSpace

ii. Method

Nama	Tipe	Parameter	Deskripsi
EquationSpace	Public konstruktor	Int independentvariable	Konstruktor untuk EquationSpace

c. Class EuclideanSpace

i. Attribute

Nama	Tipe	Deskripsi
dimension	Public int	Variable untuk menentukan dimensi dari sebuah persamaan

ii. Method

Nama	Tipe	Parameter	Deskripsi
EuclideanSpace	konstruktor	dimension	Konstruktor untuk membuat euclideanSpace
_createNewZeroVector	EuclideanSpace		Men-set nilai dari semua yang ada di dalam euclideanSpace menjadi 0

d. Class GradientEquation

i. Attribute

Nama	Tipe	Deskripsi
equationCount	Public static int	Banyaknya turunan berarah
F	Public EquationSpace	Variable untuk turunan berarah
F_x	Public EquationSpace	Variable untuk turunan berarah terhadap x
F_y	Public EquationSpace	Variable untuk turunan berarah terhadap y

F_xy	Public EquationSpace	Variable untuk turunan berarah terhadap xy
------	----------------------	--

ii. Method

Nama	Tipe	Parameter	Deskripsi
toArray	Public EquationSpace	-	Konstruktor untuk turunan berarah yang diubah ke dalam EquationSpace

e. Class VectorSpace

i. Attribute

Nama	Tipe	Deskripsi
Basiscount	Public int	Banyaknya persamaan yang ada di dalam array data
data	Double[]	Membuat array yang berisi persamaan dari hasil operasi regresi linear, interpolinom, dan bicubic spline

4. Package Point

a. Class Point

i. Attribute

Nama	Tipe	Deskripsi
x	Public double	Variable untuk nilai koordinat x
y	Public double	Variable untuk nilai koordinat y

ii. Method

Nama	Tipe	Parameter	Deskripsi
Point	Public Constructor	Double x, double y	Membuat titik kooordinat dengan komponen x dan y

5. Package CLI.IO

a. Class MainMenu

i. Attribute

Nama	Tipe	Deskripsi
scanner	Scanner	Menerima input dari user

ii. Method

Nama	Tipe	Parameter	Deskripsi
InterfaceProgram	void	-	Menampilkan seluruh menu dari program ke terminal

b. Class Prompter

i. Attribute

Nama	Tipe	Deskripsi
Scanner	Scanner	Mendapatkan input dari CLI

ii. Method

Nama	Tipe	Parameter	Deskripsi
printMultiLineString	void	String[]	Melakukan print ke CLI untuk multiline string
getEuclideanVectorInline	EuclideanSpace	String, int	Mendapatkan vektor dari CLI dengan basis parameter
getBoolean	Boolean	String	Mendapatkan boolean dari CLI
getInteger	Integer	String	Mendapatkan integer dari CLI
getDouble	Double	String	Mendapatkan double dari CLI
getBoundedInt	Integer	String, int, int	Mendapatkan integer yang dibatasi range tertentu
getString	String	String	Mendapatkan string dari CLI
get	T	String, Function<String, T>, Function<String, String>	Mendapatkan sebuah data dari CLI berdasarkan fungsi yang diberikan
waitEnter	void	-	Menunggu input dari user
getMatrix	Matrix	-	Mendapatkan matrix dari user

c. Class StringFormatter

i. Attribute

Nama	Tipe	Deskripsi
-	-	-

ii. Method

Nama	Tipe	Parameter	Deskripsi
offsetIntString	String	Value, fn	
createSubscript	String	Value	Membuat char subscript untuk beberapa persamaan
createSuperscript	String	Value	Membuat char superscript (pangkat) untuk beberapa persamaan
matrix	String	M	Mengubah hasil dari perhitungan matrix menjadi string

Commented [IJ1]: @Muhammad Atpur Rafif Ini buat apa ya mas?

bicubicSpline	String	Eq	Hasil dari bicubicSpline diubah menjadi dalam bentuk String
multipleRegression	String	Eq	Hasil dari regresi linear berganda diubah dalam bentuk string
polynomialEquation	String	Eq	Hasil dari polynomial equation diubah menjadi dalam bentuk string