

IF2123 Aljabar Linear dan Geometri
SISTEM PERSAMAAN LINEAR, DETERMINAN, DAN APLIKASINYA

Laporan Tugas Besar 1

Disusun untuk memenuhi tugas mata kuliah Aljabar Linear dan Geometri pada Semester 1 (satu)

Tahun Akademik 2024/2025



Oleh

Ahmad Syafiq	13523135
Amira Izani	13523143
Natalia Desiany Nursimin	13523157

Kelompok Nama Kelompok

PROGRAM STUDI TEKNIK INFORMATIKA
SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA
INSTITUT TEKNOLOGI BANDUNG
BANDUNG
2024

DAFTAR ISI

DAFTAR ISI	2
BAB 1 DESKRIPSI MASALAH	4
BAB 2 TEORI SINGKAT	7
2.1 Sistem Persamaan Linear	7
2.2 Determinan	7
2.3 Balikan Matriks	8
2.4 Interpolasi Polinom	8
2.5 Interpolasi Bicubic Spline	8
2.6 Regresi Linear Berganda	10
BAB 3 IMPLEMENTASI PUSTAKA	11
3.1 Class Matrix	11
3.1.1. Atribut	11
3.1.2. Metode	12
3.2 Class OperasiMatrix	12
3.2.1. Atribut	12
3.2.2. Metode	13
3.3 Class Determinan	15
3.2.1. Atribut	15
3.x.2. Metode	15
3.4 Class Invers	16
3.4.1. Atribut	16
3.4.2. Metode	16
3.5 Class SPL	17
3.5.1. Atribut	17
3.5.2. Metode	17
3.6 Class SolusiBanyak	18
3.6.1. Atribut	18
3.6.2. Metode	18
3.7 Class InterpolasiPolinomial	19
3.7.1. Atribut	19
3.7.2. Metode	19
3.8 Class RegresiBerganda	20
3.8.1. Atribut	20
3.8.2. Metode	20
3.9 Class BicubicInterpolation	22
3.9.1. Atribut	22
3.9.2. Metode	22
3.10 Class Main	23

3.10.1. Atribut	23
3.10.2. Metode	23
3.11 Class IO	26
3.11.1. Atribut	26
3.11.2. Metode	26
4 EKSPERIMEN	27
BAB 5 KESIMPULAN	36
5.1 Kesimpulan	36
5.2 Saran	36
5.3 Komentar	36
5.4 Refleksi	37
LAMPIRAN	37

BAB 1 DESKRIPSI MASALAH

Untuk memenuhi Tugas Besar 1 dalam mata kuliah Aljabar Linear dan Geometri, penulis mengembangkan sebuah pustaka aljabar linear menggunakan bahasa pemrograman Java. Pustaka ini memiliki kemampuan untuk menyelesaikan sistem persamaan linear, menghitung determinan serta matriks balikan, menyelesaikan interpolasi polinomial, dan melakukan regresi linear. Pustaka tersebut kemudian digunakan dalam program yang dapat menyelesaikan berbagai persoalan yang dimodelkan dalam bentuk sistem persamaan linear, interpolasi polinomial, dan regresi linear.

Spesifikasi program adalah sebagai berikut:

1. Program dapat menerima masukan (*input*) baik dari *keyboard* maupun membaca masukan dari *file text*. Untuk SPL, masukan dari *keyboard* adalah m , n , koefisien a_{ij} , dan b_i . Masukan dari *file* berbentuk matriks *augmented* tanpa tanda kurung, setiap elemen matriks dipisah oleh spasi. Misalnya,

```
3 4.5 2.8 10 12  
-3 7 8.3 11 -4  
0.5 -10 -9 12 0
```

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

```
3 4.5 2.8  
-3 7 8.3  
0.5 -10 -9
```

Luaran (*output*) disesuaikan dengan persoalan (determinan atau invers) dan penghitungan balikan/invers dilakukan dengan metode matriks balikan dan adjoint.

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 $(8.0, 2.0794)$, $(9.0, 2.1972)$, dan $(9.5, 2.2513)$ dan akan mencari nilai y saat $x = 8.3$, maka di dalam *file text* ditulis sebagai berikut:

```
8.0 2.0794  
9.0 2.1972  
9.5 2.2513  
8.3
```

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

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

dan untuk regresi adalah

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

untuk kasus regresi kuadratik, variabel boleh menggunakan x_1 , x_2 , dan lain-lain tetapi perlu dijelaskan variabel tersebut merepresentasikan apa. Contoh

$$x_1 = X$$

.

$$x_3 = X^2$$

.

$$x_5 = XY$$

[Persamaan dan Solusi]

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 disekitaranya, diikuti dengan nilai a dan b untuk mencari nilai $f(a, b)$.

Misalnya jika nilai dari $f(0, 0), f(1, 0), f(0, 1), f(1, 1), f_x(0, 0), f_x(1, 0), f_x(0, 1), f_x(1, 1), f_y(0, 0), f_y(1, 0), f_y(0, 1), f_y(1, 1), f_{xy}(0, 0), f_{xy}(1, 0), f_{xy}(0, 1), f_{xy}(1, 1)$ 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 dapat dibuat dengan pilihan menu. Urutan menu dan isinya dipersilakan dirancang masing-masing. Misalnya, menu:

MENU

1. Sistem Persamaan Linier
2. Determinan
3. Matriks balikan
4. Interpolasi Polinom
5. Interpolasi Bicubic Spline
6. Regresi linier dan kuadratik berganda
7. Interpolasi Gambar (Bonus)
8. 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
- 5.

Begitu juga untuk pilihan menu nomor 2, 3, dan 6

BAB 2 TEORI SINGKAT

2.1 Sistem Persamaan Linear

Sistem persamaan linear adalah kumpulan persamaan linear yang saling terkait. Persamaan linear adalah persamaan yang memiliki variabel dengan pangkat satu. Sistem persamaan linear dapat terdiri dari satu variabel, dua variabel, atau lebih.

2.1.1 Metode Eliminasi Gauss

Metode eliminasi Gauss adalah teknik matematika yang digunakan untuk menyelesaikan sistem persamaan linear dengan mengubah matriks augmented menjadi bentuk eselon, di mana setiap baris memiliki leading one (satu elemen terkemuka) di kolom tertentu dan semua elemen di bawah leading one adalah nol. Langkah-langkahnya melibatkan operasi baris seperti pertukaran baris, penggantian baris, dan pembagian baris untuk mencapai bentuk eselon ini.

2.1.2 Metode Eliminasi Gauss-Jordan

Metode eliminasi Gauss-Jordan adalah teknik matematika yang digunakan untuk menemukan solusi sistem persamaan linear dan mengubah matriks augmented menjadi bentuk eselon tereduksi, di mana setiap baris memiliki satu elemen terkemuka (leading one) di kolom tertentu, dengan semua elemen di atas dan di bawah leading one menjadi nol. Ini dilakukan melalui serangkaian operasi elemen baris, termasuk pertukaran baris, penggantian baris, dan pembagian baris.

2.1.3 Metode Matriks Balikan

Metode penentuan SPL dengan menggunakan matriks balikan hanya dapat digunakan pada matriks persegi dan juga ketika determinan $\neq 0$. Ide dasarnya adalah dengan mengalikan matriks koefisien dari sistem persamaan dengan matriks inversnya ($x = A^{-1}b$), sehingga kita mendapatkan matriks solusi yang berisi nilai-nilai variabel yang dicari.

2.1.4 Kaidah Cramer Kaidah Cramer adalah metode khusus untuk menyelesaikan sistem persamaan linear dengan variabel sebanyak yang sama dengan jumlah persamaan. Menurut Kaidah Cramer, jika $Ax = b$ adalah SPL yang terdiri dari n persamaan linier dengan n peubah (variable) sedemikian sehingga $\det(A) \neq 0$, maka SPL tersebut memiliki solusi yang unik, yaitu:

$$x_1 = \frac{\det(A_i)}{\det(A)}$$

dimana A_i adalah matriks yang diperoleh dengan mengganti entri pada kolom ke- i dari A dengan entri dari matriks b .

2.2 Determinan

Determinan adalah sebuah abstraksi yang melambangkan suatu nilai yang bisa didapatkan dari sebuah matriks persegi. Nilai determinan digunakan untuk mengukur sifat-sifat geometris dan aljabar matriks tersebut. Determinan dari suatu matriks persegi A umumnya dilambangkan dengan $\det(A)$. Proses perhitungan determinan melibatkan aturan dan metode khusus, seperti metode ekspansi kofaktor atau metode reduksi baris, tergantung pada ukuran dan sifat matriks yang bersangkutan.

2.2.1 Metode Reduksi Baris

Metode reduksi baris adalah salah satu teknik yang digunakan dalam aljabar linear untuk menyederhanakan dan mempermudah pemecahan sistem persamaan linear. Determinan matriks A

dapat diperoleh dengan melakukan OBE pada matriks A sampai diperoleh matriks segitiga (segitiga bawah atau atas).

2.2.2 Metode Ekspansi Kofaktor

Ekspansi kofaktor pada dasarnya hanya menuliskan determinan sebagai jumlah linear determinan dari satu baris/kolom, lalu menukar baris dan kolom dari setiap determinan tersebut untuk menjadikan setiap matriks segitiga atas .

2.3 Balikan Matriks

Balikan matriks, juga dikenal sebagai matriks invers, adalah matriks yang ketika dikalikan dengan matriks asalnya menghasilkan matriks identitas. Matriks balikan hanya ada jika determinan matriks asalnya tidak nol. Proses menemukan matriks balikan melibatkan teknik-teknik seperti eliminasi Gauss-Jordan atau menggunakan metode matriks Adjoin.

2.3.1 Metode Matriks Adjoin

Metode matriks adjoin adalah teknik dalam aljabar matriks yang digunakan untuk menemukan matriks adjoin dari matriks persegi. Matriks adjoin adalah matriks yang diperoleh dengan menukar elemen-elemen matriks dengan kofaktor masing-masing dan kemudian melakukan tranposisi. Balikan matriks A dapat dihitung dengan menggunakan rumus:

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

2.3.2 Metode Eliminasi Gauss Jordan

Metode eliminasi Gauss-Jordan dapat digunakan untuk menghitung matriks balikan. Misalkan A adalah matriks persegi berukuran $n \times n$. Balikan (inverse) matriks A adalah A^{-1} sedemikian sehingga $AA^{-1} = A^{-1}A = I$. Untuk matriks A yang berukuran $n \times n$, matriks balikannya, yaitu A^{-1} , dicari dengan cara berikut:

$$[A|I] \sim [I|A^{-1}]$$

2.4 Interpolasi Polinom

Interpolasi polinom merupakan teknik interpolasi dengan mengasumsikan pola data yang kita miliki mengikuti pola polinomial baik berderajat satu (linier) maupun berderajat tinggi. Interpolasi dengan metode ini dilakukan dengan terlebih dahulu membentuk persamaan polinomial $P_n(x)$ dari $n+1$ buah titik berbeda, $(x_0, y_0), (x_1, y_1), \dots, (x_n, y_n)$ sehingga $y_i = P_n(x_i)$ untuk $i = 0, 1, 2, \dots, n$. Sehingga, kita dapat menggunakan persamaan polinomial tersebut untuk menghitung perkiraan nilai y di x sembarang.

2.5 Interpolasi Bicubic Spline

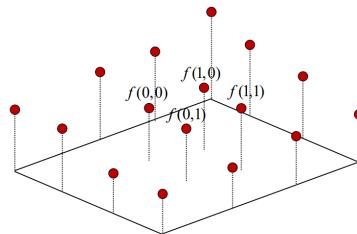
Bicubic spline interpolation adalah metode interpolasi yang digunakan untuk mengaproksimasi fungsi di antara titik-titik data yang diketahui. *Bicubic spline interpolation* melibatkan konsep *spline* dan konstruksi serangkaian polinomial kubik di dalam setiap sel segi empat dari data yang diberikan.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

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 membagun 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.

$$y = X\alpha$$

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

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 .

2.6 Regresi Linear Berganda

Regresi linear berganda matriks adalah teknik statistik yang digunakan untuk memodelkan hubungan antara variabel dependen dan satu atau lebih variabel independen dengan menggunakan matriks. Hal ini dilakukan dengan memprediksi persamaan yang berlaku menggunakan data-data yang ada hingga membentuk sebuah persamaan linear. 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 β_i dapat digunakan *Normal Estimation Equation for Multiple Linear Regression* sebagai berikut:

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

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^2
- 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!

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

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 uv . 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.

BAB 3 IMPLEMENTASI PUSTAKA

3.1 Class Matrix

3.1.1. Atribut

contents	array 2D bertipe double yang digunakan untuk menyimpan elemen matriks.
rows	integer yang menyimpan jumlah baris dalam matriks
cols	integer yang menyimpan jumlah kolom

	dalam matriks.
--	----------------

3.1.2. Metode

<code>public Matrix(int row, int col)</code>	Konstruktor Matrix
<code>public int getLastRow()</code>	Mengembalikan jumlah baris matriks
<code>public int getLastCol()</code>	Mengembalikan jumlah kolom matriks
<code>public boolean isValidRow(int row)</code>	Fungsi yang mengembalikan benar jika $0 \leq row \leq \text{rows}$
<code>public boolean isValidCol(int col)</code>	Fungsi yang mengembalikan benar jika $0 \leq col \leq \text{cols}$
<code>public void setElement(int row, int col, double val)</code>	Metode untuk mengubah elemen ke-row col
<code>public static Matrix readMatrixFromKeyboard()</code>	Metode untuk membaca matriks dari terminal.
<code>public static Matrix readMatrixFromFile(String fileName)</code>	Metode untuk membaca matriks dari sebuah file yang berada di folder test, parameter filename berisikan nama file.
<code>public static Matrix readNxNMatrixFromKeyboard()</code>	Metode untuk membaca matriks NxN dari terminal.
<code>public static Matrix readNxNMatrixFromFile(String fileName)</code>	Metode untuk membaca matriks dari sebuah file yang berada di folder test, parameter filename berisikan nama file. Metode ini hanya bisa membaca matriks NxN
<code>public void TulisMatrix()</code>	Prosedur menampilkan matriks

3.2 Class OperasiMatrix

Class ini berisikan metode operasi biner maupun uner yang dilakukan pada matriks

3.2.1. Atribut

Class ini tidak memiliki atribut

3.2.2. Metode

<pre>public static Matrix copyMatrix(Matrix m)</pre>	Mengembalikan salinan dari matrix m
<pre>public static double returnDetBySarrus(Matrix m)</pre>	Mengembalikan determinan matrix m menggunakan kaidah sarrus
<pre>public static Matrix getMatrixKoefisien(Matrix m)</pre>	Mengembalikan matrix koefisien dari matrix augmented m
<pre>public static Matrix getVektorKonstanta(Matrix m)</pre>	Mengembalikan vektor konstanta dari matrix augmented m
<pre>public static Matrix replaceColWithVector(Matrix m, Matrix vector, int col_index)</pre>	Mengembalikan matrix m yang kolom ke-col_indexnya diubah menjadi vector. Fungsi ini digunakan untuk SPL Cramer
<pre>public static double cofactorExpansion(Matrix matrix, int n)</pre>	Mengembalikan nilai determinan matrix berdimensi nxn dengan
<pre>public static void getMinor(Matrix matrix, Matrix minor, int rowToSkip, int colToSkip, int n)</pre>	Fungsi getMinor menghasilkan matriks minor dengan menghapus baris dan kolom tertentu dari matriks input. Parameter matrix adalah matriks asal, sedangkan minor adalah matriks yang dihasilkan. Fungsi ini melewati baris yang sesuai dengan rowToSkip dan kolom yang sesuai dengan colToSkip, lalu mengisi elemen-elemen lain ke dalam matriks minor. Variabel n menunjukkan ukuran matriks asal yang digunakan dalam perulangan.
<pre>public static Matrix returnInversByAdjoint(Matrix matrix)</pre>	Fungsi returnInversByAdjoint mengembalikan invers dari matriks menggunakan metode adjoint. Pertama, ia menghitung adjoint dari matriks input, dan jika hasil adjoint null, fungsi mengembalikan null. Lalu, ia menghitung determinan matriks, dan jika determinannya nol, fungsi juga mengembalikan null karena matriks tidak memiliki invers. Jika tidak, setiap elemen matriks invers dihitung dengan

	membagi elemen adjoint dengan determinan, dan matriks invers dikembalikan.
<pre>public static Matrix resultAdjoint(Matrix matrix)</pre>	Fungsi resultAdjoint menghitung dan mengembalikan matriks adjoint dari matriks input. Jika determinan matriks adalah nol, fungsi mengembalikan null. Untuk setiap elemen, fungsi menghitung minor dan kofaktor, lalu mengisi elemen adjoint dengan kofaktor yang sesuai. Matriks adjoint yang dihitung dikembalikan dalam bentuk transpose.
<pre>public static double returnDetByRowReduction(Matrix m, int rows)</pre>	Fungsi returnDetByRowReduction menghitung determinan matriks dengan reduksi baris. Jika diperlukan, baris ditukar untuk menghindari elemen diagonal nol. Setelah eliminasi, determinan dihitung dari perkalian elemen diagonal, dengan penyesuaian tanda jika ada pertukaran baris.
<pre>public static Matrix REF(Matrix m)</pre>	Fungsi REF mengubah matriks input menjadi bentuk <i>Row Echelon Form</i> (REF)
<pre>public static Matrix ReductionREF(Matrix m)</pre>	Fungsi ReductionREF mengubah matriks input menjadi bentuk <i>Reduced Row Echelon Form</i> (RREF).
<pre>public static Matrix swapTwoRows(Matrix m, int row1, int row2)</pre>	Fungsi swapTwoRows memindahkan posisi row1 dan row2 pada matrix m
<pre>public static Matrix returnInversByGaussJordan(Matrix m, int N)</pre>	Fungsi returnInversByGaussJordan mengembalikan invers dari matriks m menggunakan metode eliminasi Gauss-Jordan.
<pre>public static String[] SolveSPLGaussJordan(Matrix m)</pre>	Fungsi SolveSPLGaussJordan untuk menyelesaikan sistem persamaan linear (SPL) menggunakan metode eliminasi Gauss-Jordan.
<pre>public static Matrix transpose(Matrix matrix)</pre>	Fungsi transpose melakukan proses transpose pada matrix, yaitu mengubah

	baris menjadi kolom dan kolom menjadi baris.
<pre>public static Matrix toAugmented(Matrix XTX, Matrix vektor)</pre>	Fungsi toAugmented menggabungkan matrix XTX dengan sebuah vektor kolom untuk membentuk sebuah matriks augmented.

3.3 Class Determinan

3.2.1. Atribut

Class ini tidak memiliki atribut

3.x.2. Metode

<pre>public static void calculateDetBySarrus (Matrix m)</pre>	Fungsi calculateDetBySarrus memanggil fungsi returnDetBySarrus dari class OperasiMatrix untuk menyimpan nilai determinan menggunakan kaidah sarrus dari matrix m dan menampilkan matrix dan nilai determinan dalam terminal.
<pre>public static void calculateDetByCofactorExpansion (Matrix matrix)</pre>	Fungsi calculateDetByCofactorExpansion memanggil fungsi cofactorExpansion dari class OperasiMatrix untuk menyimpan nilai determinan menggunakan metode cofactor expansion dari matrix m dan akan menampilkan matrix dan nilai determinan dalam terminal. Apabila baris dan kolom matriks tidak sama, maka akan mengeluarkan "Determinant hanya dapat dihitung untuk matriks persegi (NxN)"
<pre>public static void calculateDetByRowReduction (Matrix matrix)</pre>	Fungsi calculateDetByRowReduction memanggil fungsi returnDetByRowReduction untuk menyimpan nilai determinan menggunakan metode reduksi baris dari matrix matrix dan akan menampilkan matrix dan nilai

	determinan dalam terminal. Apabila baris dan kolom matriks tidak sama, maka akan mengeluarkan "Determinant hanya dapat dihitung untuk matriks persegi (NxN)"
--	--

3.4 Class Invers

3.4.1. Atribut

Class ini tidak memiliki atribut

3.4.2. Metode

```
public static void
getInversByAdjoint(Matrix
matrix)
```

Fungsi getInversByAdjoint menghitung invers dari sebuah matriks menggunakan metode adjoint. Fungsi awalnya mengecek apakah matriks berbentuk persegi. Jika matriks tidak persegi, maka akan muncul pesan bahwa invers hanya bisa dihitung untuk matriks persegi dan proses berhenti. Kemudian, fungsi akan mengecek determinan melalui metode ekspansi kofaktor. Jika determinan 0, pesan akan ditampilkan bahwa matriks tidak memiliki invers, karena invers dari matriks hanya bisa dihitung jika determinan tidak sama dengan 0. Kemudian, fungsi akan menampilkan matriks awal, lalu menghitung invers dengan memanggil OperasiMatrix.returnInversByAdjoint(matrix)

```
public static void
getInversByRowReduction(Ma
trix matrix)
```

Fungsi getInversByRowReduction berfungsi untuk menghitung invers dari sebuah matriks menggunakan metode reduksi baris (Gauss-Jordan). Pertama-tama, fungsi akan memeriksa apakah matriks berbentuk persegi, lalu menggunakan metode reduksi baris OperasiMatrix.returnDetByRowReduc tion untuk menghitung determinan matriks. Jika determinan 0, pesan akan ditampilkan bahwa matriks tidak

	memiliki invers. Selanjutnya, matriks yang dimasukkan akan ditampilkan, kemudian invers akan dihitung.
--	--

3.5 Class SPL

3.5.1. Atribut

Class ini tidak memiliki atribut

3.5.2. Metode

<pre>public static void solveCramerRule(Matrix m)</pre>	Fungsi solveCramerRule ini menyimpan hasil SPL dari fungsi returnDetByRowReduction. Fungsi ini juga mengecek apakah determinan bernilai 0. Apabila bernilai 0, maka akan mengerluarkan "Determinan matriks adalah 0 .
<pre>public static void SolveInverseMatrix(Matrix matrix)</pre>	Fungsi SolveInverseMatrix ini menyelesaikan SPL menggunakan metode invers matrix.
<pre>public static Matrix SPLByInverseMatrix(Matrix matrixKoefisien, Matrix vektorKonstanta)</pre>	Fungsi SPLByInverseMatrix menghitung SPL menggunakan invers matriks. Setelah itu, mengembalikan hasil dari SPL dalam bentuk matrix.
<pre>Public static Matrix kalikanMatriks(Matrix m1, Matrix m2)</pre>	Fungsi kalikanMatriks berfungsi untuk mengalikan dua buah matrix. Setelah itu, fungsi akan mengembalikan matriks hasil perkalian dalam bentuk output.
<pre>public static void solveGaussianElimination (M atrix m)</pre>	Fungsi solveGaussianElimination berfungsi untuk menyelesaikan SPL menggunakan metode eliminasi Gauss Matrix m akan diubah menjadi bentuk REF menggunakan OperasiMatrix.REF(m). Setelah itu, solusi akan dihitung dan ditampilkan.
<pre>public static void</pre>	Fungsi

<code>solveGaussianJordanElimination(Matrix m)</code>	solveGaussianJordanElimination berfungsi untuk menyelesaikan SPL dengan menggunakan metode eliminasi Gauss-Jordan. Matrix m akan ditransformasikan ke dalam bentuk RREF. Kemudian, solusi akan dicari dan ditampilkan.
<code>public static boolean isVariable(char c)</code>	Fungsi isVariable mengecek apakah sebuah karakter adalah variabel (huruf).

3.6 Class SolusiBanyak

3.6.1. Atribut

Class ini tidak memiliki atribut

3.6.2. Metode

<code>public SolusiBanyak()</code>	Class SolusiBanyak berfungsi untuk merepresentasikan solusi dari persamaan dengan banyak variabel melalui array koefisien list_koef.
<code>public static String MergeSolusiBanyak(SolusiBanyak sb)</code>	Fungsi MergeSolusiBanyak berfungsi untuk menggabungkan elemen-elemen dari array list_koef ke dalam satu string solusi yang dapat dibaca. Metode ini mengambil objek SolusiBanyak dan menghasilkan representasi string dari solusi tersebut. Jika ada nilai independen yang tidak Kemudian, metode akan melewati setiap indeks dari array list_koef
<code>public static SolusiBanyak sbKaliKonstanta(SolusiBanyak sb, double c)</code>	Fungsi SolusiBanyaksbKaliKonstanta berfungsi untuk mengalikan seluruh koefisien dalam objek SolusiBanyak dengan konstanta c.
<code>public static SolusiBanyak subtractSolusiBanyak(SolusiBanyak sb1, SolusiBanyak</code>	Fungsi SolusiBanyaksubtractSolusiBanyak berfungsi untuk mengurangkan dua

sb2)	objek SolusiBanyak dengan mengurangi koefisien di sb2 dari sb1. Metode ini mengembalikan objek SolusiBanyak baru yang merupakan hasil pengurangan koefisien dari dua objek SolusiBanyak yang diberikan.
public static int nextIndexSB(int idx)	Fungsi nextIndexSB memberikan indeks variabel berikutnya dalam urutan tertentu. Metode ini digunakan untuk menemukan indeks berikutnya berdasarkan aturan spesifik tentang variabel yang digunakan.

3.7 Class InterpolasiPolinomial

3.7.1. Atribut

Class ini tidak memiliki atribut

3.7.2. Metode

public static void bacaKeyboardInterpolasiPolinomial()	Fungsi bacaKeyboardInterpolasiPolinomial berfungsi untuk memungkinkan pengguna untuk memasukkan titik data secara manual melalui keyboard.
public static void bacaFileInterpolasiPolinomial(String fileName)	Fungsi bacaFileInterpolasiPolinomial membaca data titik dari file yang ditentukan oleh fileName. Metode ini akan membuka file dan menghitung jumlah baris yang ada untuk menentukan jumlah titik data (n). Data x dan y kemudian diambil dari file dan disimpan dalam array. Nilai x untuk interpolasi diambil dari baris terakhir file. Setelah membaca data, metode ini membentuk matrix Vandermonde dan memanggil metode interpolate untuk melakukan interpolasi.
public static void	Fungsi interpolate berfungsi untuk

```
interpolate(Matrix V,  
Matrix vektorY, double  
xTarget)
```

menggunakan Matriks Vandermonde (V) dan vektor Y. Pertama, fungsi akan memeriksa apakah determinan matrix Vandermonde adalah 0 menggunakan metode cofactorExpansion. Jika 0, interpolasi tidak dapat dilakukan. Jika matriks memiliki invers, metode returnInversByGaussJordan akan digunakan untuk mendapatkan invers dari matriks V. Kemudian, koefisien polinomial dihitung dengan mengalikan invers V dengan vektorY. Lalu, hasil interpolasi akan dihitung dan dicetak.

```
public static Matrix  
createVandermondeMatrix (do  
uble[] x, int n)
```

Fungsi createVandermondeMatrix berfungsi untuk membuat matrix Vandermonde berdasarkan array x. Matriks V akan diinisialisasi dengan ukuran $n \times n$.

```
public static Matrix  
arrayToVector(double[] l,  
int n)
```

Fungsi arrayToVector berfungsi untuk mengubah array l menjadi sebuah vektor. Setiap elemen dari array l dimasukkan ke dalam kolom pertama dari matriks hasil..

3.8 Class RegresiBerganda

3.8.1. Atribut

Class ini tidak memiliki atribut

3.8.2. Metode

```
public static void  
regresiKuadratikBerganda (i  
nt n, int m, Scanner sc)
```

Fungsi regresiKuadratikBerganda digunakan untuk melakukan regresi kuadratik berganda. Pertama, fungsi akan membaca input untuk setiap sampel, juga menghitung kuadrat dari setiap variabel dan interaksi antar variabel. Setelah itu, fungsi akan membangun matrix dan menyelesaikan regresi, kemudian menampilkan

	koefisien dan nilai taksiran berdasarkan variabel yang diberikan oleh pengguna.
<pre>public static void bacaFileRegresiKuadratikBerganda(String fileName)</pre>	Fungsi bacaFileRegresiKuadratikBerganda membaca data untuk regresi kuadratik dari file. Pertama, fungsi akan membuka dan membaca file yang berisi data sampel. Kemudian, menghitung jumlah sampel dan membangun matriks. Setelah itu, hasil koefisien regresi akan dihitung dan akan ditampilkan taksiran berdasarkan variabel yang diambil dari baris terakhir file.
<pre>public static String[] multipleRegressionSolution (Matrix X, Matrix y, int m, int totalColumns)</pre>	Fungsi multipleRegressionSolution digunakan untuk menghitung koefisien regresi menggunakan metode regresi berganda. Pertama, transpose akan dihitung dari matriks. Hasil perkalian akan dihitung dan matriks augmented akan dibuat. SPL akan dicari menggunakan metode eliminasi Gauss-Jordan untuk menemukan solusi, yaitu nilai koefisien regresi.
<pre>public static void regresiLinierBerganda(int n, int m, Scanner sc)</pre>	Fungsi regresiLinierBerganda berfungsi untuk menghitung regresi linier berganda. Dengan jumlah variabel bebas n dan jumlah sampel m, fungsi ini menginisialisasi matriks X dan vektor y, kemudian meminta input dari pengguna untuk mengisi nilai variabel dan target. Setelah semua data diinput, fungsi ini menghitung koefisien regresi dan menampilkannya.
<pre>public static void bacaFileRegresiLinear(String fileName) throws FileNotFoundException</pre>	Fungsi bacaFileRegresiLinear bertugas untuk membuka file, membaca data, dan menyimpan informasi ke dalam matriks X dan vektor y. Setelah

	mengisi data, fungsi ini menghitung koefisien regresi menggunakan metode regresi berganda dan menampilkannya beserta taksirannya.
<pre>public static String hampiran(int n, String[] solusi, double[] variable)</pre>	Fungsi hampiran menghitung nilai taksiran berdasarkan koefisien regresi dan variabel yang diberikan. Fungsi ini menghitung taksiran berdasarkan koefisien dan variabel yang diberikan. Hasil taksiran ini kemudian dikembalikan dalam bentuk string.

3.9 Class BicubicInterpolation

3.9.1. Atribut

Class ini tidak memiliki atribut

3.9.2. Metode

<pre>public static void bacaKeyboardBicubicInterpolation()</pre>	Fungsi regresiKuadratikBerganda berfungsi membaca masukan dari keyboard untuk membangun sebuah matriks berukuran 4x4 yang terdiri dari nilai fungsional dan turunan, serta membaca nilai a dan b untuk interpolasi. Fungsi ini berisi validasi input dan logika untuk memastikan bahwa nilai a dan b berada dalam rentang (0, 1).
<pre>public static void bicubicSplineInterpolation (Matrix input, double a, double b)</pre>	Fungsi bicubicSplineInterpolation membentuk vektor y yang berisi elemen-elemen dari matriks input dan membuat matriks x berdasarkan turunan parsial dan nilai di titik-titik yang relevan. Matriks ini berukuran 16x16. Setelah itu, invers dari matriks x akan dihitung, lalu dikalikan dengan vektor y untuk mendapatkan vektor koefisien a. Kemudian, fungsi akan memanggil fungsi returnBicubicInterpolation() untuk menghitung hasil interpolasi.

<pre>public static void bacaFileBicubicSpline(Stri- ng fileName)</pre>	Fungsi bacaFileBicubicSpline membaca data dari file yang berisi matriks 4x4 serta nilai a dan b. Jika file tidak ada atau format file tidak sesuai, akan muncul pesan kesalahan.
<pre>public static double returnBicubicInterpolation (Matrix vektor_a, double x, double y)</pre>	Fungsi returnBicubicInterpolation menghitung hasil interpolasi bicubic di titik (x, y) dengan menggunakan vektor koefisien a. Koefisien ini dikalikan dengan hasil ekspansi sigma berdasarkan nilai x dan y, lalu dijumlahkan untuk memberikan hasil akhir interpolasi.
<pre>public static double ekspansiSigmaHasil(double x, double y, int i, int j)</pre>	Fungsi ekspansiSigmaHasil berfungsi untuk enghitung hasil dari ekspansi sigma untuk interpolasi di titik (x, y).
<pre>public static double ekspansiSigmaTurunanX(doub- le x, double y, int i, int j)</pre>	Fungsi ekspansiSigmaTurunanX menghitung hasil ekspansi sigma untuk turunan terhadap x di titik (x, y).
<pre>public static double ekspansiSigmaTurunanY(doub- le x, double y, int i, int j)</pre>	Fungsi ekspansiSigmaTurunanY menghitung hasil ekspansi sigma untuk turunan terhadap y.
<pre>public static double ekspansiSigmaTurunanSilang (double x, double y, int i, int j)</pre>	Fungsi ekspansiSigmaTurunanSilang menghitung hasil ekspansi sigma untuk turunan silang.

3.10 Class Main

3.10.1. Atribut

Class ini tidak memiliki atribut

3.10.2. Metode

<pre>public static void main(String[] args)</pre>	Fungsi main berfungsi sebagai titik masuk utama dalam program. Di dalam main, user dapat memilih metode-metode yang ingin digunakan.
---	--

<pre>public static void solveByGaussElimination()</pre>	Fungsi solveByGaussElimination digunakan untuk menyelesaikan sistem persamaan linier dengan metode Eliminasi Gauss.
<pre>public static void solveByGaussJordanElimination()</pre>	Fungsi solveByGaussJordanElimination digunakan untuk menyelesaikan sistem persamaan linier dengan metode Eliminasi GaussJordan.
<pre>public static void solveByInverseMatrix()</pre>	Fungsi solveByInverseMatrix digunakan untuk menyelesaikan sistem persamaan linier menggunakan metode invers matriks. Dalam metode ini, jika kita memiliki sistem persamaan linier yang dinyatakan dalam bentuk matriks sebagai $AX=B$, maka solusi X dapat ditemukan dengan mengalikan invers dari A dengan B , tetapi hanya berlaku jika determinan tidak 0.
<pre>public static void solveByCramerRule()</pre>	Fungsi solveByCramerRule digunakan untuk menyelesaikan sistem persamaan linier menggunakan Aturan Cramer.
<pre>public static void determinantByCofactorExpansion()</pre>	Fungsi determinantByCofactorExpansion digunakan untuk menyelesaikan SPL menggunakan Metode Matriks Adjoin. Solusi untuk SPL dapat ditemukan dengan mengalikan matriks adjoint dari matriks A dengan konstanta vektor B dan kemudian membaginya dengan determinan A .
<pre>public static void determinantByRowReduction()</pre>	Fungsi determinantByRowReduction akan menghitung determinan matriks dengan metode Reduksi Baris.
<pre>public static void determinantBySarrus()</pre>	Fungsi determinantBySarrus berfungsi untuk menghitung determinan menggunakan Metode Sarrus, yang hanya berlaku untuk matriks berukuran

	3x3. Jika ukuran matriks bukan 3x3, fungsi akan memberikan pesan kesalahan dan kembali meminta input. Setelah input valid, matriks dihitung menggunakan metode Determinant.calculateDetBySarrus(matrix).
<pre>public static void inversByAdjoint()</pre>	Fungsi inversByAdjoint digunakan untuk menghitung invers matriks dengan Metode Adjoint. Setelah matriks diinput, fungsi Invers.getInversByAdjoint(matrix) akan memproses perhitungan invers.
<pre>public static void inversByRowReduction()</pre>	Fungsi inversByRowReduction digunakan untuk menghitung invers matriks dengan Metode Gauss-Jordan (Reduksi Baris). Setelah matriks diinput, fungsi Invers.getInversByRowReduction(matrix) akan menghitung invers matriks.
<pre>public static void startInterpolasiPolinomial() ()</pre>	Fungsi startInterpolasiPolinomial berfungsi untuk memproses Interpolasi Polinomial.
<pre>public static void startBicubicSpline() ()</pre>	Fungsi startBicubicSpline digunakan untuk menjalankan proses Spline Bicubic. Setelah memilih metode input, fungsi memanggil BicubicInterpolation.bacaKeyboardBicubicInterpolation() atau BicubicInterpolation.bacaFileBicubicSpline(fileName) untuk membaca data dan menjalankan interpolasi bicubic.

<pre>public static void startLinearRegression()</pre>	Fungsi startLinearRegression berfungsi untuk memproses Regresi Linier Berganda. Pengguna bisa memilih antara input keyboard atau file untuk memasukkan jumlah peubah dan sampel. Setelah itu, regresi linier dihitung menggunakan RegresiBerganda.regresiLinierBerganda(n, m, sc) atau RegresiBerganda.bacaFileRegresiLinear(fileName).
<pre>public static void startQuadraticRegression()</pre>	Fungsi startQuadraticRegression memproses Regresi Kuadratik Berganda. Setelah pengguna memasukkan data input, fungsi RegresiBerganda.regresiKuadratikBerganda(n, m, sc) atau RegresiBerganda.bacaFileRegresiKuadratik(fileName) akan dijalankan untuk menghitung regresi kuadratik.
<pre>public static void delay(int ms)</pre>	Fungsi delay digunakan untuk menunda eksekusi selama ms (milidetik). Hal ini membantu memberikan jeda antar operasi seperti penundaan tampilan output di konsol.

3.11 Class IO

3.11.1. Atribut

Class ini tidak memiliki atribut

3.11.2. Metode

<pre>public static void tekanEnterUntukKembali()</pre>	Fungsi tekanEnterUntukKembali bertujuan untuk memberikan instruksi kepada pengguna agar menekan tombol Enter agar dapat kembali ke menu utama.
<pre>public static void tulisSolusiSPL(String []</pre>	Fungsi tulisSolusiSPL bertujuan untuk menampilkan solusi dari sistem

solutions)

persamaan linier (SPL). Metode ini pertama-tama memeriksa apakah solusi pertama dalam array solutions adalah "Tidak ada solusi". Jika ya, maka akan mencetak "Tidak ada solusi" di konsol. Jika ada solusi, metode ini akan mencetak solusi yang diperoleh.

BAB 4 EKSPERIMEN

4.1. Temukan solusi SPL $AX = b$

Soal		Metode	Output
a.	$A = \begin{bmatrix} 1 & 1 & -1 & -1 \\ 2 & 5 & -7 & -5 \\ 2 & -1 & 1 & 3 \\ 5 & 2 & -4 & 2 \end{bmatrix}, \quad b = \begin{bmatrix} 1 \\ -2 \\ 4 \\ 6 \end{bmatrix}$	Gauss	<pre>Masukkan nama file (contoh: matrix.txt): 1_a.txt 1.0 1.0 -1.0 -1.0 1.0 0.0 1.0 -1.6666666666666667 -1.0 -1.333333333333333 0.0 0.0 1.0 -1.0 1.0 0.0 0.0 0.0 0.0 1.0 Tidak ada solusi</pre>
		Gauss-Jordan	<pre>1.0 0.0 0.0 0.6666666666666667 1.6666666666666665 0.0 1.0 0.0 -2.6666666666666667 0.333333333333333 0.0 0.0 1.0 -1.0 1.0 0.0 0.0 0.0 0.0 1.0 Tidak ada solusi</pre>
		Balikan	<pre>Masukkan nama file (contoh: matrix.txt): 1_a.txt Matriks tidak memiliki invers karena determinan adalah 0.</pre>
		Crammer	<pre>Masukkan nama file (contoh: matrix.txt): 1_a.txt Determinan matriks adalah 0.</pre>
b.	$A = \begin{bmatrix} 1 & -1 & 0 & 0 & 1 \\ 1 & 1 & 0 & -3 & 0 \\ 2 & -1 & 0 & 1 & -1 \\ -1 & 2 & 0 & -2 & -1 \end{bmatrix}, \quad b = \begin{bmatrix} 3 \\ 6 \\ 5 \\ -1 \end{bmatrix}$	Gauss	<pre>Masukkan nama file (contoh: matrix.txt): 1_b.txt 1.0 -1.0 0.0 0.0 1.0 3.0 0.0 1.0 0.0 -1.5 -0.5 1.5 0.0 0.0 0.0 1.0 -1.0 -1.0 0.0 0.0 0.0 0.0 0.0 0.0 x1 = 3.0+r, x2 = 2.0r, x3 = s, x4 = -1.0+r, x5 = r</pre>
		Gauss-Jordan	<pre>1.0 0.0 0.0 0.0 -1.0 3.0 0.0 1.0 0.0 0.0 -2.0 0.0 0.0 0.0 0.0 1.0 -1.0 -1.0 0.0 0.0 0.0 0.0 0.0 0.0 x1 = 3.0+r, x2 = 2.0r, x3 = s, x4 = -1.0+r, x5 = r</pre>

		Balikan	N = 6
			<pre>Masukkan nama file (contoh: matrix.txt): 1_d.txt x1 = -0.400000000000000, x2 = -21.2500000000000, x3 = -11.7500000000000, x4 = 10.0, x5 = 0.000000000000000, x6 = -1.00000000000000</pre>
		N = 10	<pre>Masukkan nama file (contoh: matrix.txt): 1_d.txt x1 = -0.239359018988607, x2 = -21.5512793651507, x3 = -157.5687000150105, x4 = 20.1, x5 = 0.000000000000000, x6 = -0.755437702217043, x7 = -1.00000000000000</pre>
		Crammer	<pre>Masukkan nama file (contoh: matrix.txt): 1_d.txt x1 = -0.000000000000000, x2 = -21.0000000000000, x3 = -11.0000000000000, x4 = 10.0, x5 = 0.000000000000000, x6 = -1.00000000000000</pre>

4.2. SPL Berbentuk Matriks Augmented

Soal		Metode	Output
a.	$\begin{bmatrix} 1 & -1 & 2 & -1 & -1 \\ 2 & 1 & -2 & -2 & -2 \\ -1 & 2 & -4 & 1 & 1 \\ 3 & 0 & 0 & -3 & -3 \end{bmatrix}$	Gauss	<pre>Masukkan nama file (contoh: matrix.txt): 2_a.txt 1.0 -1.0 2.0 -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 x1 = -1.0+r, x2 = 2.0s, x3 = s, x4 = r Tekan Enter untuk kembali ke menu utama...</pre>
		Gauss-Jordan	<pre>Masukkan nama file (contoh: matrix.txt): 2_a.txt 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 x1 = -1.0+r, x2 = 2.0s, x3 = s, x4 = r</pre>
		Balikan	<pre>Masukkan nama file (contoh: matrix.txt): 2_a.txt Matriks tidak memiliki invers karena determinan adalah 0.</pre>
		Crammer	<pre>Masukkan nama file (contoh: matrix.txt): 2_a.txt Determinan matriks adalah 0.</pre>
b.	$\begin{bmatrix} 2 & 0 & 8 & 0 & 8 \\ 0 & 1 & 0 & 4 & 6 \\ -4 & 0 & 6 & 0 & 6 \\ 0 & -2 & 0 & 3 & -1 \\ 2 & 0 & -4 & 0 & -4 \\ 0 & 1 & 0 & -2 & 0 \end{bmatrix}$	Gauss	<pre>Masukkan nama file (contoh: matrix.txt): 2_b.txt 1.0 0.0 4.0 0.0 4.0 0.0 1.0 0.0 4.0 6.0 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 y1 = 0, x2 = 2.0, x3 = 1.0, x4 = 1.0</pre>

	Gauss-Jordan	<pre>Masukkan nama file (contoh: matrix.txt): 2_b.txt 1.0 0.0 0.0 0.0 0.0 0.0 1.0 0.0 0.0 2.0 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 x1 = 0, x2 = 2.0, x3 = 1.0, x4 = 1.0</pre>
	Balikan	<pre>Masukkan nama file (contoh: matrix.txt): 2_a.txt Dimensi 6x5 tidak valid untuk penyelesaian SPL dengan matriks balikan.</pre>
	Crammer	<pre>Masukkan nama file (contoh: matrix.txt): 2_b.txt Dimensi 6x5 tidak valid untuk penyelesaian SPL dengan kaidah cramer.</pre>

4.3. SPL Terbentuk

		Crammer	Masukkan nama file (contoh: matrix.txt): 3_0.txt Dimensi 12x10 tidak valid untuk penyelesaian SPL dengan kaidan cramer.

4.4. Sistem Reaktor

Soal	Metode	Output
a. A: $m_{A_{in}} + Q_{BA}x_B - Q_{AB}x_A - Q_{AC}x_A = 0$ B: $Q_{AB}x_A - Q_{BA}x_B - Q_{BC}x_B = 0$ C: $m_{C_{out}} + Q_{AC}x_A + Q_{BC}x_B - Q_{C_{out}}x_C = 0$ Tentukan solusi x_A, x_B, x_C dengan menggunakan parameter berikut: $Q_{AB} = 40, Q_{AC} = 80, Q_{BA} = 60, Q_{BC} = 20$ dan $Q_{C_{out}} = 150 \text{ m}^3/\text{s}$ dan $m_{A_{in}} = 1300$ dan $m_{C_{out}} = 200 \text{ mg/s}$.	Gauss	Masukkan nama file (contoh: matrix.txt): 4.txt 3 3 x 0.000000000000000 0.0 3 x 1.000000000000000 0.0 0.0 1.0 -0.772208139534867 0.0 x1 = 0.01r, x2 = 1.00r, x3 = 0.77r, x4 = r
	Gauss-Jordan	Masukkan nama file (contoh: matrix.txt): 4.txt 1.0 0.0 0.0 -0.008720930232558155 0.0 0.0 1.0 0.0 -1.659837209302322 0.0 0.0 0.0 1.0 -0.7732558139534862 0.0 x1 = 0.01r, x2 = 1.00r, x3 = 0.77r, x4 = r
	Balikan	Masukkan nama file (contoh: matrix.txt): 4.txt Dimensi 3x3 tidak valid untuk penyelesaian SPL dengan metrik balikan.
	Crammer	Masukkan nama file (contoh: matrix.txt): 4.txt Dimensi 3x3 tidak valid untuk penyelesaian SPL dengan kaidan cramer.

4.5. Studi Kasus Interpolasi Polinom

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

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

X = 0,2

```
Masukkan nama file (contoh: matrix.txt): 5_a_1.txt
f(x) = 0+0.2399999999997534x+0.1973958333338473x^2-9.947598300641403E-14x^3+0.026041666666969
Hasil interpolasi untuk x = 0.2 adalah: 0.033
```

X = 0,55

```
Masukkan nama file (contoh: matrix.txt): 5_a_2.txt
f(x) = 0+0.2399999999997534x+0.1973958333338473x^2-9.947598300641403E-14x^3+0.02604166666696983x^4-1.84
Hasil interpolasi untuk x = 0.55 adalah: 0.1711
```

X = 0,85

```
Masukkan nama file (contoh: matrix.txt): 5_a_3.txt
f(x) = 0+0.2399999999997534x+0.1973958333338473x^2-9.947598300641403E-14x^3+0.0260416666696983x^4-1.8474111297
Hasil interpolasi untuk x = 0.85 adalah: 0.3372
```

X = 1,28

```
Masukkan nama file (contoh: matrix.txt): 5_a_4.txt
f(x) = 0+0.2399999999997534x+0.19739583333338473x^2-9.947598300641403E-14x^3+0.0260416666696983x^4-1.8474111297
Hasil interpolasi untuk x = 1.28 adalah: 0.6775
```

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

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

Gunakanlah data di atas dengan memanfaatkan interpolasi polinomial untuk melakukan prediksi jumlah kasus baru Covid-19 pada tanggal-tanggal berikut:

- a. 16/07/2022

```
Masukkan nama file (contoh: matrix.txt): 5_b_a.txt
f(x) = 7187066071649-9.346993079177645E12x+5.3342030
Hasil interpolasi untuk x = 7.516 adalah: 53574.002
```

- b. 10/08/2022

```
Masukkan nama file (contoh: matrix.txt): 5_b_b.txt  
f(x) = 6587116812-3.785834641716499E9x+7.625426512037488E8  
Hasil interpolasi untuk x = 8.323 adalah: -401024.4876
```

c. 05/09/2022

```
Masukkan nama file (contoh: matrix.txt): 5_b_c.txt  
f(x) = 6587116812-3.785834641716499E9x+7.625426512037488E8  
Hasil interpolasi untuk x = 9.166 adalah: -4415514.101
```

d. Masukan user lainnya berupa tanggal (desimal) yang sudah diolah dengan asumsi prediksi selalu dilakukan untuk tahun 2022.

```
Masukkan nama file (contoh: matrix.txt): 5_b_d.txt  
f(x) = 6587116812-3.785834641716499E9x+7.625426512037488E8  
Hasil interpolasi untuk x = 10.3 adalah: -7.38021599925E7
```

c. Sederhanakan fungsi $f(x)$ yang memenuhi kondisi

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

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

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

$N = 4$

```
Masukkan nama file (contoh: matrix.txt): 5_c_4.txt  
f(x) = 0+1.592833333333342x-1.856166666666663x^2+1.000000000000000x^3-0.166666666666667x^4+0.000000000000000x^5  
Hasil interpolasi untuk x = 1.0 adalah: 0.538
```

$N = 5$

```
Masukkan nama file (contoh: matrix.txt): 5_c_5.txt  
f(x) = 0+2.0260000000000034x-3.525000000000028x^2+3.200000000000000x^3-1.000000000000000x^4+0.000000000000000x^5  
Hasil interpolasi untuk x = 1.0 adalah: 0.5347
```

$N = 6$

$f(x) = 0 + 2.460701165960484x - 5.03160980622336x^2 + 4.97848x^3$
 Hasil interpolasi untuk $x = 1.0$ adalah: 0.5232

4.6. Studi Kasus Regresi Linear dan Kuadratik Berganda

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.

Penyelesaian Regresi Linear Berganda untuk di atas menggunakan Normal Estimation Equation for Multiple Linear Regression untuk mendapatkan regresi linear berganda dari data pada tabel di atas, diperoleh sistem persamaan linear sebagai berikut.

$$20b_0 + 863.1b_1 + 1530.4b_2 + 587.84b_3 = 19.42$$

$$863.1b_0 + 54876.89b_1 + 67000.09b_2 + 25283.395b_3 = 779.477$$

$$1530.4b_0 + 67000.09b_1 + 117912.32b_2 + 44976.867b_3 = 1483.437$$

$$587.84b_0 + 25283.395b_1 + 44976.867b_2 + 17278.5086b_3 = 571.1219$$

Kemudian diestimasi nilai Nitrous Oxide($f(x)$) dengan Humidity bernilai 50%, temperatur 76°F, dan tekanan udara sebesar 29.30 didapatkan hasil sebagai berikut :

```

Masukkan nama file (contoh: matrix.txt): 6.txt
1.0 0.0 0.0 0.0 -3.5077781408835103
0.0 1.0 0.0 0.0 -0.002624990745878327
0.0 0.0 1.0 0.0 7.989410472218274E-4
0.0 0.0 0.0 1.0 0.15415503019830143
Koefisien Regresi β₀ = -3.51
Koefisien Regresi β₁ = 0.0
Koefisien Regresi β₂ = 0.0
Koefisien Regresi β₃ = 0.15
Hampiran (taksiran) nilai f(x): 0.8849999999999998

```

4.7. Studi Kasus Interpolasi Bicubic Spline

Diberikan matriks input dengan bentuk sebagai berikut. Format matriks masukan bukan mewakili nilai matriks, tetapi mengikuti format masukan pada bagian “Spesifikasi Tugas” nomor 7.

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

f(0, 0):

```

Masukkan nama file (contoh: matrix.txt): 7_a.txt
f(0.0,0.0) = 21.0

```

f(0.5, 0.5):

```

Masukkan nama file (contoh: matrix.txt): 7_b.txt
f(0.5,0.5) = 87.796875

```

f(0.25, 0.75) :

```
Masukkan nama file (contoh: matrix.txt): 7_c.txt  
f(0.25,0.75) = 117.732177734375
```

f(0.1, 0.9):

```
Masukkan nama file (contoh: matrix.txt): 7_d.txt  
f(0.1,0.9) = 128.57518700000003
```

BAB 5 KESIMPULAN

5.1 Kesimpulan

Pada pengerjaan proyek Tugas Besar 1 IF 2123 Aljabar Linear dan Geometri ini, kami telah membuat sebuah program yang dapat mengimplementasikan berbagai metode pengolahan matriks untuk menyelesaikan Sistem Persamaan Linier (SPL). Metode yang digunakan termasuk Metode Eliminasi Gauss, Metode Eliminasi Gauss-Jordan, Metode Matriks Balikan, dan Kaidah Cramer. Setiap metode memiliki karakteristik dan kegunaan masing-masing, sehingga program ini memberikan fleksibilitas kepada pengguna dalam memilih metode yang sesuai berdasarkan jenis matriks dan SPL yang dihadapi dan ingin dicari. Hasil solusi yang dihasilkan oleh aplikasi dapat berupa solusi unik, solusi banyak, atau bahkan tidak ada solusi, tergantung pada sifat matriks yang digunakan. Selain itu, program ini juga dilengkapi dengan fitur untuk menghitung determinan, matriks balikan, interpolasi polinom, interpolasi bicubic spline, serta regresi linier dan kuadratik berganda. Secara keseluruhan, program ini menyediakan solusi yang komprehensif dalam pengolahan matriks dan pemodelan regresi, mempermudah penyelesaian berbagai permasalahan.

5.2 Saran

1. Pemahaman Bahasa Pemrograman

Pengembangan program ini membutuhkan pemahaman yang mendalam terhadap bahasa pemrograman Java. Bagi mahasiswa yang baru mengenal Java, disarankan untuk meluangkan waktu lebih dalam memahami sintaks dan konsep dasar penggunaan Java, terutama terkait pengelolaan matriks dan operasi linier yang cukup kompleks.

2. Kerja Sama Tim

Pelaksanaan tugas ini melibatkan banyak bagian yang harus diselesaikan dengan kerja sama tim yang efektif. Penggunaan alat kolaborasi seperti Github atau platform sejenis dapat mempermudah pengelolaan kode agar sinkron dan meningkatkan efektivitas.

3. Pengembangan Fungsi Tambahan

Kedepannya, pengembangan program dapat dilakukan dengan melibatkan lebih banyak fungsi pengolahan matriks, serta kemampuan menangani sistem persamaan non-linier untuk memperluas cakupan program. Penambahan antarmuka pengguna yang lebih interaktif juga bisa dipertimbangkan agar program memiliki daya tarik lebih.

5.3 Komentar

Secara keseluruhan, proyek ini cukup menantang dan lumayan sulit untuk dikerjakan, namun memberikan wawasan mendalam tentang penerapan teori aljabar linier dalam pengembangan perangkat lunak. Meskipun terdapat banyak hambatan yang harus diatasi, seperti kesulitan teknis dalam pemrograman dan penerapan algoritma yang rumit, proyek ini memberikan pengalaman belajar yang sangat berharga.

5.4 Refleksi

Tugas ini telah membantu kami bertiga mengasah kemampuan dalam memahami konsep-konsep dasar aljabar linier dan pemrograman Java. Meskipun awalnya menghadapi beberapa kendala teknis dalam implementasi algoritma dan pengolahan matriks, serta kesulitan dalam mempelajari pemrograman Java. Namun, melalui kolaborasi tim dan usaha dalam mendalami materi lebih lanjut, kami berhasil menyelesaikan tantangan-tantangan yang ada. Dari sini, kami menyadari pentingnya keterampilan problem-solving serta kemampuan kerja tim yang solid untuk menghadapi proyek yang kompleks. Pengalaman ini sangat berharga dalam membekali kami untuk menghadapi tantangan serupa dan proyek-proyek lainnya di masa yang akan datang.

LAMPIRAN

6.1 Referensi

Rinaldi Munir. "Aljabar dan Geometri untuk Informatika 2023/2024." informatika.stei.itb.ac.id/~rinaldi.munir/AljabarGeometri/2023-2024/algeo23-24.html

Rowe, John B. "Bicubic Interpolation of Digital Images."
https://www.mssc.mu.edu/~daniel/pubs/RoweTalkMSCS_BiCubic.pdf

6.2 Tautan Repository

<https://github.com/iammadsfq/Algeo01-23135>