

LAPORAN TUGAS BESAR 1

IF2123 ALJABAR LINIER DAN GEOMETRI



Disusun Oleh:

Kelompok 41 - tapi maaf bgt apa boleh sya atur algeo e smean

Naufarrel Zhafif Abhista (13523149)

Hasri Fayadh Muqaffa (13523156)

I Made Wiweka Putera (13523160)

SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA
INSTITUT TEKNOLOGI BANDUNG
SEMESTER 1 TAHUN 2024/2025

DAFTAR ISI

DAFTAR ISI.....	2
BAB I	
Deskripsi Masalah.....	4
BAB II	
Teori Singkat.....	8
1. Sistem Persamaan Linier.....	8
2. Operasi Baris Elementer (OBE).....	8
3. Eliminasi Gauss.....	9
4. Eliminasi Gauss-Jordan.....	9
5. Kaidah Cramer.....	9
6. Matriks Kofaktor dan Adjoin.....	9
7. Determinan Matriks.....	11
8. Matriks Balikan.....	11
9. Interpolasi Polinom.....	12
10. Regresi Linear Berganda.....	13
11. Bicubic Spline Interpolation.....	15
12. Image Resizing and Stretching.....	18
BAB III	
Implementasi Pustaka dan Program Java.....	20
3.1 com.smean.kalkulatorsmean.....	20
3.1.1 Determinan.java.....	20
3.1.2 ImageResizing.java.....	21
3.1.3 InterpolasiBicubicSpline.java.....	22
3.1.4 InterpolasiPolinom.java.....	23
3.1.5 IOMatriks.java.....	23
3.1.6 Main.java.....	24
3.1.7 MatriksBalikan.java.....	24
3.1.8 OBE.java.....	25
3.1.9 RegresiBerganda.java.....	27
3.1.10 SPL.java.....	28
3.1.11 Parametrik.java.....	29
BAB IV	
Eksperimen.....	31
1. Test Case Studi Kasus.....	31
BAB V	
Kesimpulan.....	60
Kesimpulan.....	60
Saran.....	60
Refleksi.....	60

BAB I

Deskripsi Masalah

1. Buatlah 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. Gunakan pustaka di atas untuk membuat program penyelesaian berbagai persoalan dalam bentuk SPL, menyelesaikan persoalan interpolasi dan regresi, menghitung matriks balikan, menghitung determinan matriks dengan berbagai metode (reduksi baris dan ekspansi kofaktor).

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 invers, metode yang digunakan ada 2 yaitu menggunakan OBE dan Matriks Adjoin
4. 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

5. 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.
6. 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 $x_1 = t$).
7. 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]

8. Untuk persoalan *bicubic spline interpolation*, masukan dari *file text* (.txt) yang berisi matriks berukuran 4×4 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 $f(0, 0), f(1, 0), f(0, 1), f(1, 1), f_x(0, 0), f_x(1, 0), f_x(0, 1), f_x(1, 1), f_y(0, 0), f_y(1, 0), f_y(0, 1), f_y(1, 1), f_{xy}(0, 0), f_{xy}(1, 0), f_{xy}(0, 1), f_{xy}(1, 1)$ 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 $f(0.5, 0.5)$.

9. Luaran program harus dapat ditampilkan pada layar komputer dan dapat disimpan ke dalam *file*.
10. Bahasa program yang digunakan adalah Java. Anda bebas untuk menggunakan versi java apapun dengan catatan di atas java versi 8 (8/9/11/15/17/19/20/21).

11. 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

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

BAB II

Teori Singkat

1. Sistem Persamaan Linier

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

$$\left[\begin{array}{cccc} 0 & 2 & 1 & -1 \\ 0 & 0 & 3 & 1 \\ 0 & 0 & 0 & 0 \end{array} \right] \sim \left[\begin{array}{cccc} 0 & 1 & 0 & -\frac{2}{3} \\ 0 & 0 & 1 & \frac{1}{3} \\ 0 & 0 & 0 & 0 \end{array} \right].$$

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

2. Operasi Baris Elementer (OBE)

Operasi Baris Elementer (OBE) adalah serangkaian teknik yang digunakan dalam aljabar linear untuk melakukan manipulasi pada baris-baris matriks. OBE sering digunakan untuk menyelesaikan sistem persamaan linier, menemukan invers matriks, dan menentukan determinan. Terdapat tiga jenis OBE, yaitu:

- 1) Pertukaran baris, menukar posisi dua baris dalam matriks
- 2) Perkalian baris, mengalikan semua elemen dalam sebuah baris dengan suatu bilangan bukan nol
- 3) Penjumlahan baris, menambahkan kelipatan dari suatu baris ke baris lainnya

3. Eliminasi Gauss

Eliminasi Gauss adalah proses yang melibatkan serangkaian operasi baris elementer untuk mengubah matriks augmented dari sistem persamaan linier menjadi bentuk eselon (row echelon form) atau bentuk eselon tereduksi (reduced row echelon form). Setelah itu, solusi dapat ditemukan dengan back substitution.

4. Eliminasi Gauss-Jordan

Eliminasi Gauss-Jordan adalah metode yang digunakan untuk menyelesaikan sistem persamaan linier dengan mengubah matriks augmentednya menjadi bentuk eselon tereduksi (reduced row echelon form). Metode ini merupakan perpanjangan dari eliminasi Gauss dan memungkinkan kita untuk menemukan solusi sistem persamaan dengan lebih langsung dan efisien.

5. Kaidah Cramer

Kaidah Cramer adalah metode dalam aljabar linear yang digunakan untuk menyelesaikan sistem persamaan linier dengan jumlah variabel yang sama dengan jumlah persamaan. Metode ini memanfaatkan determinan matriks untuk menemukan solusi unik dari sistem persamaan tersebut. Syarat penggunaan kaidah *Cramer* adalah:

- 1) Sistem persamaan harus memiliki solusi unik
- 2) Jumlah persamaan harus sama dengan jumlah variabel
- 3) Determinan dari matriks koefisien tidak boleh sama dengan nol

6. Matriks Kofaktor dan Adjoin

Dalam aljabar linear, matriks kofaktor dan adjoin merupakan konsep penting yang berkaitan dengan determinan dan invers matriks. Mari kita bahas satu per satu:

- **Minor Matriks**

Minor dari suatu elemen pada matriks adalah determinan submatriks yang diperoleh dengan menghilangkan baris dan kolom yang bersesuaian dengan elemen tersebut.

Misal kita punya matriks A:

$$A = \begin{vmatrix} 1 & 2 \\ 3 & 4 \end{vmatrix}$$

Minor dari elemen 2 (baris 1, kolom 2) adalah determinan dari submatriks yang tersisa setelah menghapus baris 1 dan kolom 2, yaitu nilai 3.

- **Kofaktor Matriks**

Kofaktor adalah minor suatu elemen dikalikan dengan $(-1)^{\text{jumlah indeks baris} + \text{indeks kolom}}$.

Kofaktor dari elemen a_{ij} adalah $(-1)^{i+j} * M_{ij}$ di mana:

M_{ij} adalah minor dari elemen a_{ij} , i adalah indeks baris, j adalah indeks kolom

Kofaktor dapat bernilai positif atau negatif tergantung pada posisi elemen dalam matriks.

- **Matriks Kofaktor**

Matriks kofaktor adalah matriks yang elemen-elemennya adalah kofaktor dari matriks aslinya.

- **Adjoin Matriks**

Adjoin dari suatu matriks adalah transpos dari matriks kofaktornya.

7. Determinan Matriks

Determinan adalah suatu fungsi yang mengambil matriks persegi dan menghasilkan angka yang menggambarkan beberapa sifat dari matriks tersebut, seperti apakah matriks tersebut invertible (dapat dibalik) dan seberapa besar matriks tersebut mengubah volume ruang ketika diinterpretasikan sebagai transformasi linier. Untuk matriks A berukuran $n \times n$, notasi untuk determinan sering ditulis sebagai $\det(A)$ atau $|A|$.

8. Matriks Balikan

Matriks balikan adalah matriks yang jika dikalikan dengan matriks asalnya menghasilkan matriks identitas. Matriks identitas adalah matriks persegi yang memiliki elemen diagonal utama bernilai 1 dan elemen lainnya bernilai 0. Misalkan A adalah matriks persegi (matriks dengan jumlah baris sama dengan jumlah kolom), maka inversnya dilambangkan dengan A^{-1} . Terdapat 2 metode untuk mencari matriks balikan:

- Metode determinan dan adjoint dengan persamaan sebagai berikut:

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

Keterangan:

A^{-1} = Matriks balikan dari A

$\det(A)$ = determinan matriks A , $\det(A) \neq 0$

$\text{adj}(A)$ = adjoin matriks A

- Metode Gauss-Jordan dengan operasi sebagai berikut

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

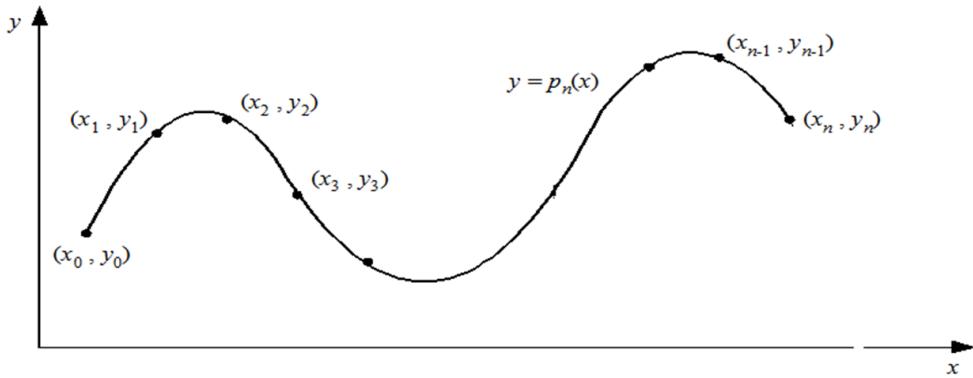
Keterangan:

\sim = Operasi Gauss – Jordan (OBE)

Metode ini menggabungkan matriks A dengan matriks identitas dalam sebuah matriks augmented, kemudian mengaplikasikan metode Gauss-Jordan hingga matriks A berubah menjadi matriks identitas dan matriks identitas menjadi matriks balikan dari matriks A .

9. Interpolasi Polinom

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



Gambar x. Ilustrasi beberapa titik yang diinterpolasi secara polinomial.

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

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

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

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

...

...

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

Solusi sistem persamaan lanjar ini, yaitu nilai a_0, a_1, \dots, a_n , diperoleh dengan menggunakan metode eliminasi Gauss yang sudah anda pelajari. Sebagai contoh, misalkan diberikan tiga buah titik yaitu $(8.0, 2.0794)$, $(9.0, 2.1972)$, dan $(9.5, 2.2513)$.

Tentukan polinom interpolasi kuadratik lalu estimasi nilai fungsi pada $x = 9.2$. Polinom kuadratik berbentuk $p_2(x) = a_0 + a_1x + a_2x^2$. Dengan menyulihkan ketiga buah titik data ke dalam polinom tersebut, diperoleh sistem persamaan lanjar yang terbentuk adalah

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

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

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

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

10. Regresi Linear Berganda

Regresi merupakan salah satu metode untuk memprediksi nilai selain menggunakan interpolasi polinom. Terdapat 2 jenis regresi yang akan digunakan, yaitu regresi linier berganda dan regresi kuadratik berganda. Kedua model regresi yang dijadikan sistem persamaan linier berikut bisa diselesaikan dengan menggunakan metode eliminasi Gauss.

1) Regresi Linier Berganda

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

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

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

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

2) Regresi Kuadratik Berganda

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

- a. Variabel Linier: Variabel dengan derajat satu seperti X, Y, dan Z
- b. Variabel Kuadrat: Variabel dengan derajat dua seperti X^2
- c. 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 bertumbuh lebih besar dibandingkan regresi linier berganda tetapi solusi tetap bisa didapat dengan menggunakan SPL.

11. Bicubic Spline Interpolation

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

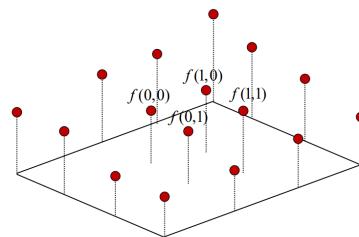
Dalam pemrosesan menggunakan interpolasi *bicubic spline* digunakan 16 buah titik, 4 titik referensi utama di bagian pusat, dan 12 titik di sekitarnya sebagai aproksimasi turunan dari keempat titik referensi untuk membagun permukaan bikubik. Bentuk pemodelannya adalah sebagai berikut.

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 x. Pemodelan interpolasi *bicubic spline*.

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

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

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

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

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

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

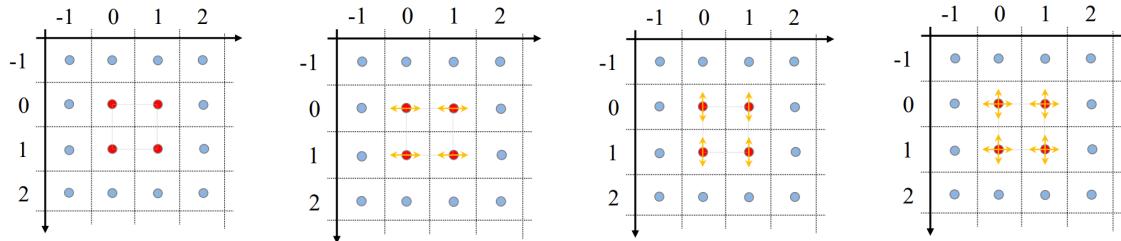
$$y = Xa$$

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

Perlu diketahui bahwa elemen pada matriks X adalah nilai dari setiap komponen koefisien a_{ij} yang diperoleh dari persamaan fungsi maupun persamaan turunan yang telah dijelaskan sebelumnya. Sebagai contoh, elemen matriks X pada baris 8 kolom ke 2 adalah koefisien dari a_{10} pada ekspansi sigma untuk $f_x(1, 1)$ sehingga diperoleh nilai konstanta $1 \times 1^{1-1} \times 1^0 = 1$, sesuai dengan isi matriks X .

Nilai dari vektor a dapat dicari dari persamaan $y = Xa$, lalu vektor a tersebut digunakan sebagai nilai variabel dalam $f(x, y)$, sehingga terbentuk fungsi interpolasi bicubic

sesuai model. Tugas Anda pada studi kasus ini adalah membangun persamaan $f(x, y)$ yang akan digunakan untuk melakukan interpolasi berdasarkan nilai $f(a, b)$ dari masukan matriks 4×4 . Nilai masukan a dan b berada dalam rentang $[0, 1]$. Nilai yang akan diinterpolasi dan turunan berarah disekitarnya dapat diilustrasikan pada titik berwarna merah pada gambar di bawah.



Gambar x. Nilai fungsi yang akan di interpolasi pada titik merah, turunan berarah terhadap sumbu x , terhadap sumbu y , dan keduanya (kiri ke kanan).

12. Image Resizing and Stretching

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

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

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

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

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

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

$$y = DI$$

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

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



Gambar x. Sebuah citra gambar asal (kiri) dan hasil pemrosesan gambar dengan skala 1.5 pada *width* dan skala 2 pada *height* (kanan).

BAB III

Implementasi Pustaka dan Program Java

Pada tugas besar kali ini, kami membuat 1 package. Package tersebut berisi fungsi-fungsi yang akan digunakan untuk mencari dan mengolah data input serta mengeluarkan output dengan format yang diinginkan.

3.1 com.smean.kalkulatorsmean

3.1.1 Determinan.java

Pada class ini dikumpulkan fungsi-fungsi yang berhubungan untuk mencari determinan dari matriks.

Fungsi/Prosedur	Deskripsi
getDeterminan (double[][] matrix, String function)	Mengembalikan nilai determinan matriks sesuai dengan metode yang dipilih oleh user
determinanOBE (double[][] matrix)	Mengembalikan nilai determinan matriks menggunakan operasi OBE
Determinankofaktor (double[][] matrix)	Mengembalikan nilai determinan matriks menggunakan operasi ekspansi kofaktor
handleInput (Scanner scanner, String function)	Menerima input sesuai pilihan user, yaitu dari keyboard atau file. Kemudian, memanggil fungsi getDeterminan untuk mendapatkan nilai determinan.

3.1.2 ImageResizing.java

Fungsi/Prosedur	Deskripsi
loadImage (String filePath)	Memuat gambar dari path yang diberikan.
resizeImageWithScale (BufferedImage	Mengubah ukuran gambar berdasarkan

image, double scaleWidth, double scaleHeight)	skala lebar (width) dan skala tinggi (height) yang diberikan menggunakan <i>bicubic interpolation</i>
bicubicInterpolationForImage (BufferedImage image, double x, double y)	Melakukan <i>bicubic interpolation</i> pada gambar untuk mendapatkan <i>pixel</i> baru pada koordinat (x, y)
calculateCoefficients (double[] matrix)	Menghitung koefisien <i>bicubic interpolation</i> berdasarkan inputan dari matriks
handleInput (Scanner scanner)	Menangani input dari pengguna untuk membuat gambar, meminta skala lebar (width) dan tinggi (height), mengubah ukuran gambar, dan menyimpan gambar yang telah diubah (resizing) ke folder “test”
bicubicInterpolation (double[][] coeffs, double a, double b)	Menghitung nilai <i>bicubic interpolation</i> berdasarkan koefisien dan posisi relatif dalam sel

3.1.3 InterpolasiBicubicSpline.java

Fungsi/Prosedur	Deskripsi
calculateCoefficients (double[] matrix)	Mengembalikan koefisien a_{00} hingga a_{33}
handleInput(Scanner scanner)	Menerima input sesuai pilihan user, yaitu dari keyboard atau file. Kemudian, memanggil fungsi calculateCoefficients dan menggunakan hasil fungsi

	calculateCoefficients beserta a dan b sebagai argumen dalam memanggil fungsi processInterpolation untuk mendapatkan nilai interpolasi $f(a, b)$.
createMatrixX()	Membuat matriks A, yaitu matriks 16x16 berisi ekspansi fungsi dari a_{00} hingga a_{33} menggunakan normalized points $f(0, 0)$, $f(1, 0)$, $f(0, 1)$, dan $f(1, 1)$.
bicubicInterpolation(double[][] coeffs, double a, double b)	Mengembalikan nilai interpolasi $f(a, b)$ setelah didapatkan koefisien, bersama dengan a dan b dari input user.

3.1.4 InterpolasiPolinom.java

Fungsi/Prosedur	Deskripsi
interpolationSolution (double[][] pointMatrix)	Mengembalikan koefisien dari fungsi interpolasi dalam matriks double.
polynomialInterpolation (double[][] pointMatrix, double x)	Mengembalikan tuple matriks. Pertama adalah koefisien dari fungsi interpolasi dalam matriks double, kedua adalah hasil substitusi interpolasi.
handleInput(Scanner scanner)	Menerima <i>input user</i> bentuk input (<i>keyboard/file</i>).
processInterpolation (double[][] setOfPoints, double x, Scanner scanner)	Memproses <i>input user</i> dan mengeluarkan bentuk persamaan interpolasi, solusinya, hasil substitusi nilai yang ingin diketahui, serta pilihan untuk menyimpan ke <i>file</i> .

3.1.5 IOMatriks.java

Fungsi/Prosedur	Deskripsi
convertTextToMatrix (String text)	Mengembalikan matriks yang berisi elemen-elemen yang dipisahkan oleh spasi dan newline dari String
convertMatrixToText (double[][] matrix)	Mengembalikan String yang berisi elemen-elemen dari sebuah matriks
readFile (String filename)	Mengembalikan matriks yang dibaca dari sebuah file
writeMatrix (double[][] matrix)	Menuliskan matriks ke terminal
getUserInput(Scanner scanner)	Menerima user input dari terminal dan mengkonversikannya menjadi sebuah matriks
saveToFile (String text, Scanner scanner)	Menyediakan opsi untuk menyimpan hasil operasi (berupa String) ke suatu file

3.1.6 Main.java

Fungsi/Prosedur	Deskripsi
clearConsole()	Menghapus semua teks yang sedang ditampilkan di terminal

3.1.7 MatriksBalikan.java

Fungsi/Prosedur	Deskripsi

normalizeMatrix (double[][] matrix)	Mengembalikan matriks yang elemen -nya disubstitusi dengan 0
getInvers (double[][] matrix, String function)	Mengembalikan matriks balikan dengan memanggil fungsi invers sesuai dengan metode yang dipilih oleh user
inversBalikan (double[][] matrix)	Mengembalikan matriks balikan dengan metode OBE dan Gauss-Jordan
inversAdjoin (double[][] matrix)	Mengembalikan matriks balikan dengan metode determinan dan adjoin
multiplyByCoef (double[][] matrix, double koefisien)	Mengembalikan matriks yang telah dikalikan dengan konstanta koefisien
getMatriksKofaktor (double[][] matrix)	Mengembalikan matriks kofaktor
getKofaktor (double[][] matrix, int x, int y)	Mengembalikan nilai kofaktor pada kolom dan baris tertentu
handleInput(Scanner scanner, String function)	Menerima input sesuai pilihan user, yaitu dari keyboard atau file. Kemudian, memanggil fungsi getInvers untuk mendapatkan matriks balikan.

3.1.8 OBE.java

Fungsi/Prosedur	Deskripsi
isSquare(double[][] matrix)	Mengembalikan true apabila matriks merupakan matriks persegi, dan false apabila matriks bukan merupakan matriks persegi.

isInversable(double[][] matrix)	Mengembalikan true apabila matriks dapat diinvers, yaitu matriks merupakan matriks persegi dan determinan matriks tidak sama dengan 0.
addMatrix(double[][] matrix1, double[][] matrix2)	Mengembalikan matriks yang merupakan hasil tambah dua buah matriks
subtractMatrix (double[][] matrix1,, double[][] matrix2)	Mengembalikan matriks yang merupakan hasil kurang dua buah matriks
copyMatrix (double[][] matrix)	Mengembalikan matriks yang merupakan salinan suatu matriks
multiplyMatrix (double[][] matrix1, double[][] matrix2)	Mengembalikan matriks yang merupakan hasil kali dua buah matriks.
transpose(double[][] matrix)	Mengembalikan matriks yang merupakan hasil transpose suatu matriks
toAugmented (double[][] squareMatrix, double[][] rhs)	Mengembalikan matriks yang merupakan augmented matriks dari matriks persegi dan matriks right hand side.
splitMatrix (double[][] augmentedMatrix)	Mengembalikan matriks tiga dimensi berisi matriks left hand side dan matriks right hand side dari suatu matriks augmented
rowMultiply (double[][] matrix, int targetedRow, int targetedCol)	Mengalikan baris dengan suatu konstanta non-zero
rowSubtract (double[][] matrix, int pivotRow, int targetedCol)	Mengurangi suatu baris dengan baris atau kelipatan baris lain

rowSwap (double[][] matrix, int rowA, int rowB)	Menukar baris a dengan baris b
nonZeroRowCheck (double[][] matrix, int pivotRow, int col)	Mengembalikan indeks baris yang tidak sama dengan 0
toRowEchelon (double[][] matrix)	Mengembalikan matriks yang merupakan matriks row echelon form menggunakan operasi OBE
toReducedRowEchelon (double[][] matrix)	Mengembalikan matriks yang merupakan matriks reduced row echelon form menggunakan operasi OBE

3.1.9 RegresiBerganda.java

Fungsi/Prosedur	Deskripsi
factorial (int n)	Mengembalikan integer dengan nilai senilai $n!$ dengan prinsip rekursi
toLinearX (double[][] independentVar)	Mengembalikan matriks double yang di-augment dengan nilai $[1 \dots 1]^T$ supaya bisa diselesaikan dengan prinsip regresi linear.
toQuadraticX (double[][] independentVar)	Mengembalikan matriks double yang di-augment dengan nilai $[1 \dots 1]^T$ dan digabungkan dengan nilai variabel awal yang dikuadratkan, serta nilai variable interaksi supaya bisa diselesaikan dengan prinsip regresi kuadratik.
convertSearchX (double[] searchX)	Mengembalikan matriks double yang

	berelemen nilai x yang akan dicari, disesuaikan dengan substitusi variabelnya nanti (di variabel linear, kuadratik, maupun interaksi).
multipleRegressionSolution (double[][] X, double[][] Y)	Mengembalikan matriks double berukuran solusi x 1.
multipleRegressionSolver (double[][] augmentedData, double[] searchX)	Mengembalikan tuple matriks. Matriks pertama adalah hasil dari multipleRegressionSolution (double[][] X, double[][] Y), matriks kedua adalah hasil substitusi solusi ke persamaan regresi.
handleInput (Scanner scanner)	Menerima <i>input user</i> berupa pilihan regresi dan bentuk input (<i>keyboard/file</i>).
processRegression (double[][] x, double[][] y, double[] searchX, Scanner scanner, int regressionChoice)	Memproses <i>input user</i> dan mengeluarkan bentuk persamaan regresi, solusinya, hasil substitusi nilai yang ingin diketahui, serta pilihan untuk menyimpan ke <i>file</i> .

3.1.10 SPL.java

Fungsi/Prosedur	Deskripsi
BackSubstitution(double[][] matrix)	Melakukan algoritma substitusi belakang KHUSUS untuk matriks dengan solusi unik.
gauss(double[][] matrix)	Mengembalikan matriks double berupa hasil SPL dari backSubstitution dengan matriks yang telah diubah ke REF.

gaussJordan(double[][] matrix)	Mengembalikan matriks double berupa hasil SPL dari backSubstitution dengan matriks yang telah diubah ke RREF.
matriksBalikan(double[][] matrix)	Mengembalikan matriks double berupa hasil SPL dari invers.
cramer(double[][] matrix)	Mengembalikan matriks double berupa hasil SPL dari metode cramer.
hasLeadingOne(double[][] matrix, int col)	Melakukan cek <i>leading one</i> untuk kolom yang dituju, <i>true</i> jika benar dan <i>false</i> jika tidak memiliki <i>leading one</i> .
checkSolution(double[][], OBEmatrix)	Cek kemungkinan solusi (Banyak, unik, tidak ada).
parametrikBackSub(double[][] OBEmatrix)	Melakukan <i>back substitution</i> KHUSUS untuk SPL dengan solusi banyak (Gauss/Gauss-Jordan)
handleInput(Scanner scanner)	Menerima <i>input user</i> berupa pilihan regresi dan bentuk input (<i>keyboard/file</i>).
processSPL(double[][] setOfPoints, Scanner scanner, int methodChoice)	Memproses <i>input user</i> dan mengeluarkan bentuk persamaan regresi, solusinya, hasil substitusi nilai yang ingin diketahui, serta pilihan untuk menyimpan ke <i>file</i> .

3.1.11 Parametrik.java

Fungsi/Prosedur	Deskripsi
Parametrik()	Membentuk tipe bentukan baru bernama

	Parametrik berupa array of double.
makeVar (Parametrik sb)	Mengubah tipe bentukan Parametrik ke string. Jika elemen parametrik bernilai 0, nilainya dihapus. Jika ada nilainya, akan digabung dengan abjad yang bersesuaian.
multiplyConstant (Parametrik sb, double k)	Mengalikan elemen parametrik dengan suatu konstanta k.
subtractParametrik (Parametrik sb1, Parametrik sb2)	Mengurangkan sb2 dengan sb1.
nextIndexParametrik (int idx)	Berpindah ke abjad selanjutnya dan digunakan nanti, bila terdapat <i>free variable</i> baru.

BAB IV

Eksperimen

1. Test Case Studi Kasus

1) Temukan solusi SPL $Ax = b$, berikut:

a.

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

Hasil:

SPL

Masukkan m (jumlah baris): 4

Masukkan n (jumlah kolom): 5

Masukkan elemennya.

a00: 1 1 -1 1 2 5 -7 -5 -2 2 -1 1 3 4 5 2 -4 2 6

a01: a02: a03: b0: a10: a11: a12: a13: b1: a20: a21: a22: a23: b2: a30: a31: a32: a33: b3:

1,000 0,000 0,000 0,667 1,667

0,000 1,000 0,000 -2,667 0,333

0,000 0,000 1,000 -1,000 1,000

0,000 0,000 0,000 0,000 1,000

Tidak ada solusi.

Simpan Hasil ke File?

1. Ya

2. Tidak

Masukkan pilihan:

SPL

Masukkan m (jumlah baris): 4

Masukkan n (jumlah kolom): 5

Masukkan elemennya.

a00: 1 1 -1 1 2 5 -7 -5 -2 2 -1 1 3 4 5 2 -4 2 6

a01: a02: a03: b0: a10: a11: a12: a13: b1: a20: a21: a22: a23: b2: a30: a31: a32: a33: b3:

Determinan matriks sama dengan 0, tidak bisa menggunakan metode invers.

SPL

Masukkan m (jumlah baris): 4

Masukkan n (jumlah kolom): 5

Masukkan elemennya.

a00: 1 1 -1 -1 2 5 -7 -5 -2 2 -1 1 3 4 5 2 -4 2 6

a01: a02: a03: b0: a10: a11: a12: a13: b1: a20: a21: a22: a23: b2: a30: a31: a32: a33: b3:

Determinan matriks sama dengan 0, tidak bisa menggunakan metode cramer.

b.

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

Hasil:

SPL

Masukkan m (jumlah baris): 4

Masukkan n (jumlah kolom): 6

Masukkan elemennya.

a00: 1 -1 0 0 1 3 1 1 0 -3 0 6 2 -1 0 1 -1 5 -1 2 0 -2 -1 -1

a01: a02: a03: a04: b0: a10: a11: a12: a13: a14: b1: a20: a21: a22: a23: a24: b2: a30: a31:

a32: a33: a34: b3:

1,000 -1,000 0,000 0,000 1,000 3,000
0,000 1,000 0,000 -1,500 -0,500 1,500
0,000 0,000 0,000 1,000 -1,000 -1,000
0,000 0,000 0,000 0,000 0,000 0,000

Banyak solusi, solusi parametrik:

x1 = 3.0+s

x2 = 2.0s

x3 = -1.0+r

x4 = -1.0+s

x5 = s

SPL

Masukkan m (jumlah baris): 4

Masukkan n (jumlah kolom): 6

Masukkan elemennya.

a00: 1 -1 0 0 1 3 1 1 0 -3 0 6 2 -1 0 1 -1 5 -1 2 0 -2 -1 -1

a01: a02: a03: a04: b0: a10: a11: a12: a13: a14: b1: a20: a21: a22: a23: a24: b2: a30: a31:

a32: a33: a34: b3:

1,000 0,000 0,000 0,000 -1,000 3,000
0,000 1,000 0,000 0,000 -2,000 0,000
0,000 0,000 0,000 1,000 -1,000 -1,000
0,000 0,000 0,000 0,000 0,000 0,000

Banyak solusi, solusi parametrik:

x1 = 3.0+s

x2 = 2.0s

x3 = -1.0+r

x4 = -1.0+s

x5 = s

○ SPL

Masukkan m (jumlah baris): 4

Masukkan n (jumlah kolom): 6

Masukkan elemennya.

a00: 1 -1 0 0 1 3 1 1 0 -3 0 6 2 -1 0 1 -1 5 -1 2 0 -2 -1 -1

a01: a02: a03: a04: b0: a10: a11: a12: a13: a14: b1: a20: a21: a22: a23: a24: b2: a30: a31: a32: a33: a34: b3:

❖ Matriks koefisien tidak persegi, tidak bisa menggunakan metode invers.

Masukkan m (jumlah baris): 4

Masukkan n (jumlah kolom): 6

Masukkan elemennya.

a00: 1 -1 0 0 1 3 1 1 0 -3 0 6 2 -1 0 1 -1 5 -1 2 0 -2 -1 -1

a01: a02: a03: a04: b0: a10: a11: a12: a13: a14: b1: a20: a21: a22: a23: a24: b2: a30: a31: a32: a33: a34: b3:

❖ Matriks koefisien tidak persegi, tidak bisa menggunakan metode cramer.

c.

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

Hasil:

o SPL

```
Masukkan m (jumlah baris): 3
Masukkan n (jumlah kolom): 7
Masukkan elemennya.
a00: 0 1 0 0 1 0 2 0 0 0 1 1 0 -1 0 1 0 0 0 1 1
a01: a02: a03: a04: a05: b0: a10: a11: a12: a13: a14: a15: b1: a20: a21: a22: a23: a24: a25:
b2:
❖ 0,000 1,000 0,000 0,000 1,000 0,000 2,000
  0,000 0,000 0,000 1,000 1,000 0,000 -1,000
  0,000 0,000 0,000 0,000 -1,000 1,000 -1,000
Banyak solusi, solusi parametrik:
x1 = 2.0+r
x2 = 3.0+t
x3 = -1.0+s
x4 = t
x5 = -1.0-t
x6 = t
```

o SPL

```
Masukkan m (jumlah baris): 3
Masukkan n (jumlah kolom): 7
Masukkan elemennya.
a00: 0 1 0 0 1 0 2 0 0 0 1 1 0 -1 0 1 0 0 0 1 1
a01: a02: a03: a04: a05: b0: a10: a11: a12: a13: a14: a15: b1: a20: a21: a22: a23: a24: a25:
b2:
❖ 0,000 1,000 0,000 0,000 1,000 0,000 2,000
  0,000 0,000 0,000 1,000 1,000 0,000 -1,000
  0,000 0,000 0,000 0,000 -1,000 1,000 -1,000
Banyak solusi, solusi parametrik:
x1 = 2.0+r
x2 = 3.0+t
x3 = -1.0+s
x4 = t
x5 = -1.0-t
x6 = t
```

o SPL

```
Masukkan m (jumlah baris): 3
Masukkan n (jumlah kolom): 7
Masukkan elemennya.
a00: 0 1 0 0 1 0 2 0 0 0 1 1 0 -1 0 1 0 0 0 1 1
a01: a02: a03: a04: a05: b0: a10: a11: a12: a13: a14: a15: b1: a20: a21: a22: a23: a24: a25:
b2:
❖ Matriks koefisien tidak persegi, tidak bisa menggunakan metode invers.
```

o SPL

```
Masukkan m (jumlah baris): 3
Masukkan n (jumlah kolom): 7
Masukkan elemennya.
a00: 0 1 0 0 1 0 2 0 0 0 1 1 0 -1 0 1 0 0 0 1 1
a01: a02: a03: a04: a05: b0: a10: a11: a12: a13: a14: a15: b1: a20: a21: a22: a23: a24: a25:
b2:
❖ Matriks koefisien tidak persegi, tidak bisa menggunakan metode cramer.
```

d.

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

H adalah matriks *Hilbert*. Cobakan untuk $n = 6$ dan $n = 10$.

Untuk $n = 6$, maka

Hasil:

SPL

Masukkan m (jumlah baris): 6

Masukkan n (jumlah kolom): 7

Masukkan elemennya.

a00: 1 0,5 0,333 0,25 0,2 0,166 1
a01: 0,25 0,2 0,166 0,142 0 0,333 0,25
a02: 0,166 0,142 0 0,333 0,25 0,2 0,166 0,14
a03: 0,125 0 0,25 0,2 0,166 0,142 0,125 0,111 0
a04: 0,111 0 0,2 0,166 0,142 0,125 0,111 0,1
a05: 0,1 0 0,166 0,142 0,125 0,111 0,111 0
b0: 0,1 0,090 0
a10: a11: a12: a13: a14: a15: b1: a20: a21: a22: a23: a24: a25:
b2: a30: a31: a32: a33: a34: a35: b3: a40: a41: a42: a43: a44: a45: b4: a50: a51: a52: a53:
a54: a55: b5:
1,000 0,500 0,333 0,250 0,200 0,166 1,000
0,000 1,000 1,006 0,904 0,795 0,711 -6,024
0,000 0,000 1,000 1,429 1,762 2,029 33,284
0,000 0,000 0,000 1,000 1,920 1,057 -31,583
0,000 0,000 0,000 0,000 1,000 0,478 10,769
0,000 0,000 0,000 0,000 0,000 1,000 67,564

Solusi:

x1 = 3.8873410102349713

x2 = -3.5415363317751414

x3 = 22.15914123103127

x4 = -61.61020874078832

x5 = -21.54312376140831

x6 = 67.56391604336102

SPL

Masukkan m (jumlah baris): 6

Masukkan n (jumlah kolom): 7

Masukkan elemennya.

a00: 1 0,5 0,333 0,25 0,2 0,166 1
a01: a02: a03: a04: a05: b0: a10: 0,5 0,333 0,25 0,2 0,166 0,142 0
0,333 0,25 0,2 0,166 0,142 0,125 0
a11: a12: a13: a14: a15: b1: a20: a21: a22: a23: a24: a25: b2: a30: 0,25 0,2 0,166 0,142 0,1
25 0,111 0
a31: a32: a33: a34: a35: b3: a40: 0,2 0,166 0,142 0,125 0,111 0,1 0
a41: a42: a43: a44: a45: b4: a50: 0,166 0,142 0,125 0,111 0,1 0,090 0
a51: a52: a53: a54: a55: b5:
1,000 0,000 0,000 0,000 0,000 0,000 3,887
0,000 1,000 0,000 0,000 0,000 0,000 -3,542
0,000 0,000 1,000 0,000 0,000 0,000 22,159
0,000 0,000 0,000 1,000 0,000 0,000 -61,610
0,000 0,000 0,000 0,000 1,000 0,000 -21,543
0,000 0,000 0,000 0,000 0,000 1,000 67,564

Solusi:

x1 = 3.8873410102349677

x2 = -3.54153633177512

x3 = 22.159141231031242

x4 = -61.610208740788316

x5 = -21.54312376140831

x6 = 67.56391604336102

SPL

Masukkan m (jumlah baris): 6

Masukkan n (jumlah kolom): 7

Masukkan elemennya.

a00: 1 0,5 0,333 0,25 0,2 0,166 1

a01: a02: a03: a04: a05: b0: a10: 0,5 0,333 0,25 0,2 0,166 0,142 0
0,333 0,25 0,2 0,166 0,142 0,125 0

a11: a12: a13: a14: a15: b1: a20: a21: a22: a23: a24: a25: b2: a30: 0,25 0,2 0,166 0,142 0,1
25 0,111 0

a31: a32: a33: a34: a35: b3: a40: 0,2 0,166 0,142 0,125 0,111 0,1 0

a41: a42: a43: a44: a45: b4: a50: 0,166 0,142 0,125 0,111 0,1 0,090 0

a51: a52: a53: a54: a55: b5:

Determinan matriks sama dengan 0, tidak bisa menggunakan metode invers.

SPL

Masukkan m (jumlah baris): 6

Masukkan n (jumlah kolom): 7

Masukkan elemennya.

a00: 1 0,5 0,333 0,25 0,2 0,166 1

a01: a02: a03: a04: a05: b0: a10: 0,5 0,333 0,25 0,2 0,166 0,142 0
0,333 0,25 0,2 0,166 0,142 0,125 0

a11: a12: a13: a14: a15: b1: a20: a21: a22: a23: a24: a25: b2: a30: 0,25 0,2 0,166 0,142 0,1
25 0,111 0

a31: a32: a33: a34: a35: b3: a40: 0,2 0,166 0,142 0,125 0,111 0,1 0

a41: a42: a43: a44: a45: b4: a50: 0,166 0,142 0,125 0,111 0,1 0,090 0

a51: a52: a53: a54: a55: b5:

Determinan matriks sama dengan 0, tidak bisa menggunakan metode cramer.

Untuk n = 10, maka

Hasil:

```

a31: a32: a33: a34: a35: a36: a37: a38: a39: b3: a40: 0,2 0,166 0,142 0,125 0,111 0,1 0,090
0,083 0,076 0,071 0
a41: a42: a43: a44: a45: a46: a47: a48: a49: b4: a50: 0,166 0,142 0,125 0,111 0,1 0,090 0,08
3 0,076 0,071 0,066 0
a51: a52: a53: a54: a55: a56: a57: a58: a59: b5: a60: 0,142 0,125 0,111 0,1 0,090 0,083 0,07
6 0,071 0,066 0,062 0
a61: a62: a63: a64: a65: a66: a67: a68: a69: b6: a70: 0,125 0,111 0,1 0,090 0,083 0,076 0,07
1 0,066 0,062 0,058 0
↳ a71: a72: a73: a74: a75: a76: a77: a78: a79: b7: a80: 0,111 0,1 0,090 0,083 0,076 0,071 0,06
6 0,062 0,058 0,055 0
a81: a82: a83: a84: a85: a86: a87: a88: a89: b8: a90: 0,1 0,090 0,083 0,076 0,071 0,066 0,06
2 0,058 0,055 0,052 0
a91: a92: a93: a94: a95: a96: a97: a98: a99: b9:
    1,000   0,500   0,333   0,250   0,200   0,166   0,142   0,125   0,111   0,100   1,000
    0,000   1,000   1,006   0,904   0,795   0,711   0,651   0,584   0,536   0,482   -6,024
    0,000   0,000   1,000   1,429   1,762   2,029   1,838   1,876   1,619   1,852   33,284
    0,000   0,000   0,000   1,000   1,920   1,057   1,760   0,947   2,478   1,030   -31,583
    0,000   0,000   0,000   0,000   1,000   0,478   1,064   -0,083   1,095   0,024   10,769
    0,000   0,000   0,000   0,000   0,000   1,000   -0,441   0,041   -1,254   -0,252   67,564
    0,000   0,000   0,000   0,000   0,000   0,000   1,000   0,793   1,750   1,053   -35,943
    0,000   0,000   0,000   0,000   0,000   0,000   0,000   1,000   0,201   0,836   -31,789
    0,000   0,000   0,000   0,000   0,000   0,000   0,000   0,000   1,000   -0,118   -21,556
    0,000   0,000   0,000   0,000   0,000   0,000   0,000   0,000   0,000   1,000   -64,832

```

Solusi:

```

x1 = 9.654473723554434
x2 = -32.468681490984864
x3 = 15.613301922633042
x4 = 2.890653430035357
x5 = -37.56820349834336
x6 = 40.35423082519431
x7 = 60.99701815605892
x8 = 28.280778726944426
x9 = -29.195256606950583
x10 = -64.83187928778135

```

a31: a32: a33: a34: a35: a36: a37: a38: a39: b3: a40: 0,2 0,166 0,142 0,125 0,111 0,1 0,090
 0,083 0,076 0,071 0
 a41: a42: a43: a44: a45: a46: a47: a48: a49: b4: a50: 0,166 0,142 0,125 0,111 0,1 0,090 0,083
 0,076 0,071 0,066 0
 a51: a52: a53: a54: a55: a56: a57: a58: a59: b5: a60: 0,142 0,125 0,111 0,1 0,090 0,083 0,076
 0,071 0,066 0,062 0
 a61: a62: a63: a64: a65: a66: a67: a68: a69: b6: a70: 0,125 0,111 0,1 0,090 0,083 0,076 0,071
 0,066 0,062 0,058 0
 a71: a72: a73: a74: a75: a76: a77: a78: a79: b7: a80: 0,111 0,1 0,090 0,083 0,076 0,071 0,066
 0,062 0,058 0,055 0
 a81: a82: a83: a84: a85: a86: a87: a88: a89: b8: a90: 0,1 0,090 0,083 0,076 0,071 0,066 0,062
 0,058 0,055 0,052 0
 a91: a92: a93: a94: a95: a96: a97: a98: a99: b9:
 1,000 0,000 0,000 0,000 0,000 0,000 0,000 0,000 0,000 0,000 0,000 0,000 9,654
 0,000 1,000 0,000 0,000 0,000 0,000 0,000 0,000 0,000 0,000 0,000 0,000 -32,469
 0,000 0,000 1,000 0,000 0,000 0,000 0,000 0,000 0,000 0,000 0,000 0,000 15,613
 0,000 0,000 0,000 1,000 0,000 0,000 0,000 0,000 0,000 0,000 0,000 0,000 2,891
 0,000 0,000 0,000 0,000 1,000 0,000 0,000 0,000 0,000 0,000 0,000 0,000 -37,568
 0,000 0,000 0,000 0,000 0,000 1,000 0,000 0,000 0,000 0,000 0,000 0,000 40,354
 0,000 0,000 0,000 0,000 0,000 0,000 1,000 0,000 0,000 0,000 0,000 0,000 60,997
 0,000 0,000 0,000 0,000 0,000 0,000 0,000 1,000 0,000 0,000 0,000 0,000 28,281
↳ 0,000 0,000 0,000 0,000 0,000 0,000 0,000 0,000 1,000 0,000 0,000 0,000 -29,195
 0,000 0,000 0,000 0,000 0,000 0,000 0,000 0,000 0,000 0,000 1,000 -64,832

Solusi:

x1 = 9.654473723554432
 x2 = -32.468681490984864
 x3 = 15.61330192263305
 x4 = 2.8906534300353712
 x5 = -37.568203498343365
 x6 = 40.3542308251943
 x7 = 60.99701815605892
 x8 = 28.280778726944426
 x9 = -29.195256606950583
 x10 = -64.83187928778135

SPL

Masukkan m (jumlah baris): 10

Masukkan n (jumlah kolom): 11

Masukkan elemennya.

a00: 1 0,5 0,333 0,25 0,2 0,166 0,142 0,125 0,111 0,1 1

a01: a02: a03: a04: a05: a06: a07: a08: a09: b0: a10: 0,5 0,333 0,25 0,2 0,166 0,142 0,125 0,111 0,1 0,090 0

a11: a12: a13: a14: a15: a16: a17: a18: a19: b1: a20: 0,333 0,25 0,2 0,166 0,142 0,125 0,111 0,1 0,090 0,083 0

a21: a22: a23: a24: a25: a26: a27: a28: a29: b2: a30: 0,25 0,2 0,166 0,142 0,125 0,111 0,1 0,090 0,083 0,076 0

a31: a32: a33: a34: a35: a36: a37: a38: a39: b3: a40: 0,2 0,166 0,142 0,125 0,111 0,1 0,090 0,083 0,076 0,071 0

a41: a42: a43: a44: a45: a46: a47: a48: a49: b4: a50: 0,166 0,142 0,125 0,111 0,1 0,090 0,083 0,076 0,071 0,066 0

↳ a51: a52: a53: a54: a55: a56: a57: a58: a59: b5: a60: 0,142 0,125 0,111 0,1 0,090 0,083 0,076 0,071 0,066 0,062 0

a61: a62: a63: a64: a65: a66: a67: a68: a69: b6: a70: 0,125 0,111 0,1 0,090 0,083 0,076 0,071 0,066 0,062 0,058 0

a71: a72: a73: a74: a75: a76: a77: a78: a79: b7: a80: 0,111 0,1 0,090 0,083 0,076 0,071 0,066 0,062 0,058 0,055 0

a81: a82: a83: a84: a85: a86: a87: a88: a89: b8: a90: 0,1 0,090 0,083 0,076 0,071 0,066 0,062 0,058 0,055 0,052 0

a91: a92: a93: a94: a95: a96: a97: a98: a99: b9:

Determinan matriks sama dengan 0, tidak bisa menggunakan metode invers.

SPL

Masukkan m (jumlah baris): 10

Masukkan n (jumlah kolom): 11

Masukkan elemennya.

a00: 1 0,5 0,333 0,25 0,2 0,166 0,142 0,125 0,111 0,1 1
a01: a02: a03: a04: a05: a06: a07: a08: a09: b0: a10: 0,5 0,333 0,25 0,2 0,166 0,142 0,125 0,111 0,1 0,090 0
a11: a12: a13: a14: a15: a16: a17: a18: a19: b1: a20: 0,333 0,25 0,2 0,166 0,142 0,125 0,111 0,1 0,090 0,083 0
a21: a22: a23: a24: a25: a26: a27: a28: a29: b2: a30: 0,25 0,2 0,166 0,142 0,125 0,111 0,1 0,090 0,083 0,076 0
a31: a32: a33: a34: a35: a36: a37: a38: a39: b3: a40: 0,2 0,166 0,142 0,125 0,111 0,1 0,090 0,083 0,076 0,071 0
a41: a42: a43: a44: a45: a46: a47: a48: a49: b4: a50: 0,166 0,142 0,125 0,111 0,1 0,090 0,083 0,076 0,071 0,066 0
a51: a52: a53: a54: a55: a56: a57: a58: a59: b5: a60: 0,142 0,125 0,111 0,1 0,090 0,083 0,076 0,071 0,066 0,062 0
a61: a62: a63: a64: a65: a66: a67: a68: a69: b6: a70: 0,125 0,111 0,1 0,090 0,083 0,076 0,071 0,066 0,062 0,058 0
a71: a72: a73: a74: a75: a76: a77: a78: a79: b7: a80: 0,111 0,1 0,090 0,083 0,076 0,071 0,066 0,062 0,058 0,055 0
a81: a82: a83: a84: a85: a86: a87: a88: a89: b8: a90: 0,1 0,090 0,083 0,076 0,071 0,066 0,062 0,058 0,055 0,052 0
a91: a92: a93: a94: a95: a96: a97: a98: a99: b9:
Determinan matriks sama dengan 0, tidak bisa menggunakan metode cramer.

2) SPL berbentuk matriks *augmented*

a.

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

Hasil:

```

...      [x] java - Algeo01-23149 + ⌂ ⌂ ... ⌂ ×
SPL

Masukkan m (jumlah baris): 4
Masukkan n (jumlah kolom): 5
Masukkan elemennya.
a00: 1 -1 2 -1 -1
2 1 -2 -2 -2
-1 2 -4 1 1
3 0 0 -3 -3
a01: a02: a03: b0: a10: a11: a12: a13: b1: a2
0: a21: a22: a23: b2: a30: a31: a32: a33: b3:

1.000 -1.000 2.000 -1.000 -1.000
0.000 1.000 -2.000 0.000 0.000
0.000 0.000 0.000 0.000 0.000
0.000 0.000 0.000 0.000 0.000

Banyak solusi, solusi parametrik:
x1 = -1.0+s
x2 = 2.0r
x3 = r
x4 = s

Simpan Hasil ke File?
1. Ya
2. Tidak
Masukkan pilihan: ■

```

```

...      [x] java - Algeo01-23149 + ⌂ ⌂ ... ⌂ ×
SPL

Masukkan m (jumlah baris): 4
Masukkan n (jumlah kolom): 5
Masukkan elemennya.
a00: 1 -1 2 -1 -1
2 1 -2 -2 -2
-1 2 -4 1 1
3 0 0 -3 -3
a01: a02: a03: b0: a10: a11: a12: a13: b1: a2
0: a21: a22: a23: b2: a30: a31: a32: a33: b3:

1.000 0.000 0.000 -1.000 -1.000
0.000 1.000 -2.000 0.000 0.000
0.000 0.000 0.000 0.000 0.000
0.000 0.000 0.000 0.000 0.000

Banyak solusi, solusi parametrik:
x1 = -1.0+s
x2 = 2.0r
x3 = r
x4 = s

Simpan Hasil ke File?
1. Ya
2. Tidak
Masukkan pilihan: ■

```

```

...      [x] java - Algeo01-23149 + ⌂ ⌂ ... ⌂ ×
SPL

Masukkan m (jumlah baris): 4
Masukkan n (jumlah kolom): 5
Masukkan elemennya.
a00: 1 -1 2 -1 -1
2 1 -2 -2 -2
-1 2 -4 1 1
3 0 0 -3 -3
a01: a02: a03: b0: a10: a11: a12: a13: b1: a2
0: a21: a22: a23: b2: a30: a31: a32: a33: b3:

Determinan matriks sama dengan 0, tidak bisa menggunakan metode invers.

Simpan Hasil ke File?
1. Ya
2. Tidak
Masukkan pilihan: ■

```

```

...      [x] java - Algeo01-23149 + ⌂ ⌂ ... ⌂ ×
SPL

Masukkan m (jumlah baris): 4
Masukkan n (jumlah kolom): 5
Masukkan elemennya.
a00: 1 -1 2 -1 -1
2 1 -2 -2 -2
-1 2 -4 1 1
3 0 0 -3 -3
a01: a02: a03: b0: a10: a11: a12: a13: b1: a2
0: a21: a22: a23: b2: a30: a31: a32: a33: b3:

Determinan matriks sama dengan 0, tidak bisa menggunakan metode cramer.

Simpan Hasil ke File?
1. Ya
2. Tidak
Masukkan pilihan: ■

```

b.

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

Hasil:

```
PROBLEMS 52 TERMINAL ... java - Algeo01-23149 + ⌂ ⌂ ... ^

Masukkan m (jumlah baris): 6
Masukkan n (jumlah kolom): 5
Masukkan elemennya.
a00: 2 0 8 0 8
0 1 0 4 6
-4 0 6 0 6
0 -2 0 3 -1
a01: a02: a03: b0: a10: a11: a12: a13: b1: a20: a21: a22: a23
: b2: a30: a31: a32: a33: b3: a40: 2 0 -4 0 -4
0 1 0 -2 0
a41: a42: a43: b4: a50: a51: a52: a53: b5:
    2.000      0.000     8.000      0.000      0.000      0.000     8.000
    0.000      1.000     0.000      4.000      0.000      0.000      6.000
   -4.000      0.000     6.000      0.000      0.000      0.000      6.000
    0.000     -2.000     0.000      3.000      0.000      0.000     -1.000
    2.000      0.000     -4.000      0.000      0.000      0.000     -4.000
    0.000      1.000     0.000     -2.000      0.000      0.000      0.000
Soluksi:
x1 = 0
x2 = 2.0
x3 = 1.0
x4 = 1.0

Simpan Hasil ke File?
1. Ya
2. Tidak
Masukkan pilihan: 1
```

```
PROBLEMS 52 TERMINAL ... ☐ java -Algeo01-23149 + ⌂ ⌂ ... ^ x
Masukkan m (jumlah baris): 6
Masukkan n (jumlah kolom): 5
Masukkan elemennya.
a00: 2 0 8 0 8
0 1 0 4 6
-4 0 6 0 6
0 -2 0 3 -1
a01: a02: a03: b0: a10: a11: a12: a13: b1: a20: a21: a22: a23
: b2: a30: a31: a32: a33: b3: a40: 2 0 -4 0 -4
0 1 0 -2 0
a41: a42: a43: b4: a50: a51: a52: a53: b5:
    2.000      0.000     8.000     0.000     0.000     0.000     0.000
    0.000      1.000     0.000     4.000     0.000     0.000     6.000
   -4.000     0.000     6.000     0.000     0.000     0.000     6.000
    0.000     -2.000     0.000     3.000     0.000     0.000     -1.000
    2.000     0.000     -4.000     0.000     0.000     0.000     -4.000
    0.000     1.000     0.000     -2.000     0.000     0.000     0.000
SoLusi:
x1 = 0
x2 = 2.0
x3 = 1.0
x4 = 1.0

Simpan Hasil ke File?
1. Ya
2. Tidak
Masukkan pilihan: ■
```

```
PROBLEMS 52 TERMINAL ... ☒ java - Algeo01-23149 + ⚏ ☐ ... ^ X

SPL

Masukkan m (jumlah baris): 6
Masukkan n (jumlah kolom): 5
Masukkan elemennya.
a00: 2 0 8 0 8
0 1 0 4 6
-4 0 6 0 6
0 -2 0 3 -1
a01: a02: a03: b0: a10: a11: a12: a13: b1: a20: a21: a22: a23
: b2: a30: a31: a32: a33: b3: a40: 2 0 -4 0 -4
0 1 0 -2 0
a41: a42: a43: b4: a50: a51: a52: a53: b5:
Matriks koefisien tidak persegi, tidak bisa menggunakan metode invers.

Simpan Hasil ke File?
1. Ya
2. Tidak
Masukkan pilihan: █
```

3) SPL berbentuk

a.

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

Hasil:

```

SPL

Masukkan m (jumlah baris): 4
Masukkan n (jumlah kolom): 5
Masukkan elemennya.
a00: 8 1 3 2 0 2 9 -1 -2 1 1 3 2 -1 2 1 0 6 4 3
a01: a02: a03: b0: a10: a11: a12: a13: b1: a20: a21: a22: a23: b2: a30: a31: a32: a33: b3:
      1,000   0,125   0,375   0,250   0,000
      0,000   1,000   -0,200   -0,286   0,114
      0,000   0,000   1,000   -0,195   0,760
      0,000   0,000   0,000   1,000   -0,258
Solusi:
x1 = -0.2243243243243243
x2 = 0.18243243243243246
x3 = 0.7094594594594594
x4 = -0.258108108108108

```

```

SPL

Masukkan m (jumlah baris): 4
Masukkan n (jumlah kolom): 5
Masukkan elemennya.
a00: 8 1 3 2 0 2 9 -1 -2 1 1 3 2 -1 2 1 0 6 4 3
a01: a02: a03: b0: a10: a11: a12: a13: b1: a20: a21: a22: a23: b2: a30: a31: a32: a33: b3:
      1,000   0,000   0,000   0,000   -0,224
      0,000   1,000   0,000   0,000   0,182
      0,000   0,000   1,000   0,000   0,709
      0,000   0,000   0,000   1,000   -0,258
Solusi:
x1 = -0.2243243243243243
x2 = 0.18243243243243246
x3 = 0.7094594594594594
x4 = -0.258108108108108

```

```

SPL

Masukkan m (jumlah baris): 4
Masukkan n (jumlah kolom): 5
Masukkan elemennya.
a00: 8 1 3 2 0 2 9 -1 -2 1 1 3 2 -1 2 1 0 6 4 3
a01: a02: a03: b0: a10: a11: a12: a13: b1: a20: a21: a22: a23: b2: a30: a31: a32: a33: b3:
Hasil penyelesaian dengan metode Cramer adalah:
x0 = -0.2243243243243243
x1 = 0.18243243243243246
x2 = 0.7094594594594594
x3 = -0.2581081081081081

```

3. Invers ($x = A^{-1} * B$)

```

SPL

Masukkan m (jumlah baris): 4
Masukkan n (jumlah kolom): 5
Masukkan elemennya.
a00: 8 1 3 2 0 2 9 -1 -2 1 1 3 2 -1 2 1 0 6 4 3
a01: a02: a03: b0: a10: a11: a12: a13: b1: a20: a21: a22: a23: b2: a30: a31: a32: a33: b3:
Hasil penyelesaian dengan metode Invers adalah:
x1 = -0.22432432432428
x2 = 0.1824324324324324
x3 = 0.7094594594594593
x4 = -0.25810810810810814

```

b.

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

Hasil:

```

SPL
a91: a92: a93: a94: a95: a96: a97: a98: b9: a100: 0,91421 0,25 0 0,25 0,91421 0,25 0 0,25 0,91421 16,13
a101: a102: a103: a104: a105: a106: a107: a108: b10: a110: 0,04289 0 0 0,75 0,04289 0 0,61396 0,75 0,04289 7,04
a111: a112: a113: a114: a115: a116: a117: a118: b11:
    0,000 0,000 0,000 0,000 0,000 0,000 1,000 1,000 1,000 0,000 0,000 0,000 0,000 0,000 13,000
    0,000 0,000 0,000 1,000 1,000 1,000 0,000 0,000 0,000 0,000 0,000 0,000 0,000 0,000 15,000
    1,000 1,000 0,000 0,000 0,000 0,000 0,000 0,000 0,000 0,000 0,000 0,000 0,000 0,000 8,000
    0,000 0,000 0,043 0,000 0,043 0,750 0,043 0,750 0,614 0,000 0,000 0,000 0,000 0,000 14,790
    0,000 0,250 0,914 0,250 0,914 0,250 0,914 0,250 0,000 0,000 0,000 0,000 0,000 0,000 0,000 14,310
    0,614 0,750 0,043 0,750 0,043 0,000 0,043 0,000 0,000 0,000 0,000 0,000 0,000 0,000 3,810
    0,000 0,000 1,000 0,000 0,000 1,000 0,000 0,000 1,000 0,000 0,000 0,000 0,000 0,000 18,000
    0,000 1,000 0,000 0,000 1,000 0,000 0,000 1,000 0,000 0,000 0,000 0,000 0,000 0,000 12,000
    1,000 0,000 0,000 1,000 0,000 0,000 0,000 1,000 0,000 0,000 0,000 0,000 0,000 0,000 6,000
    0,043 0,750 0,614 0,000 0,043 0,750 0,000 0,000 0,000 0,043 0,000 0,000 0,000 0,000 10,510
    0,914 0,250 0,000 0,250 0,914 0,250 0,000 0,250 0,914 0,000 0,000 0,000 0,000 0,000 16,130
    0,043 0,000 0,000 0,750 0,043 0,000 0,614 0,750 0,043 0,000 0,000 0,000 0,000 0,000 0,000 7,040
Tidak ada solusi.

```

```

SPL
Masukkan m (jumlah baris): 12
Masukkan n (jumlah kolom): 10
Masukkan elemennya.
a00: 0 0 0 0 0 0 1 1 1 13
0 0 0 1 1 1 0 0 0 15
a01: a02: a03: a04: a05: a06: a07: a08: b0: a10: a11: a12: a13: a14: a15: a16: a17: a18: b1: a20: 1 1 1 0 0 0 0 0 0 8
a21: a22: a23: a24: a25: a26: a27: a28: b2: a30: 0 0,04289 0 0,04289 0,75 0,04289 0,75 0,61396 14,79
a31: a32: a33: a34: a35: a36: a37: a38: b3: a40: 0 0,25 0,91421 0,25 0,91421 0,25 0,91421 0,25 0 14,31
a41: a42: a43: a44: a45: a46: a47: a48: b4: a50: 0,61396 0,75 0,04289 0,75 0,04289 0 0 3,81
0 0 0 1 0 0 0 1 1 18
a51: a52: a53: a54: a55: a56: a57: a58: b5: a60: a61: a62: a63: a64: a65: a66: a67: a68: b6: a70: 0 1 0 0 1 0 0 0 1 12
1 0 0 1 0 0 1 0 0 6
a71: a72: a73: a74: a75: a76: a77: a78: b7: a80: a81: a82: a83: a84: a85: a86: a87: a88: b8: a90: 0,04289 0,75 0,61396 0 0,04289 0,75 0 0 0,04289 10,51
a91: a92: a93: a94: a95: a96: a97: a98: b9: a100: 0,91421 0,25 0 0,25 0,91421 0,25 0 0,25 0,91421 16,13
a101: a102: a103: a104: a105: a106: a107: a108: b10: a110: 0,04289 0 0 0,75 0,04289 0 0,61396 0,75 0,04289 7,04
a111: a112: a113: a114: a115: a116: a117: a118: b11:
    0,000 0,000 0,000 0,000 0,000 1,000 1,000 1,000 0,000 0,000 0,000 0,000 0,000 13,000
    0,000 0,000 0,000 1,000 1,000 1,000 0,000 0,000 0,000 0,000 0,000 0,000 0,000 15,000
    1,000 1,000 0,000 0,000 0,000 0,000 0,000 0,000 0,000 0,000 0,000 0,000 0,000 8,000
    0,000 0,000 0,043 0,000 0,043 0,750 0,043 0,750 0,614 0,000 0,000 0,000 0,000 0,000 14,790
    0,000 0,250 0,914 0,250 0,914 0,250 0,914 0,250 0,000 0,000 0,000 0,000 0,000 0,000 14,310
    0,614 0,750 0,043 0,750 0,043 0,000 0,043 0,000 0,000 0,000 0,000 0,000 0,000 0,000 3,810
    0,000 0,000 1,000 0,000 0,000 1,000 0,000 0,000 1,000 0,000 0,000 0,000 0,000 0,000 18,000
    0,000 1,000 0,000 0,000 1,000 0,000 0,000 1,000 0,000 0,000 0,000 0,000 0,000 0,000 12,000
    1,000 0,000 0,000 1,000 0,000 0,000 1,000 0,000 0,000 0,000 0,000 0,000 0,000 0,000 6,000
    0,043 0,750 0,614 0,000 0,043 0,750 0,000 0,000 0,000 0,043 0,000 0,000 0,000 0,000 10,510
    0,914 0,250 0,000 0,250 0,914 0,250 0,000 0,250 0,914 0,000 0,000 0,000 0,000 0,000 16,130
    0,043 0,000 0,000 0,750 0,043 0,000 0,614 0,750 0,043 0,000 0,000 0,000 0,000 0,000 0,000 7,040
Tidak ada solusi.

```

```

SPL

Masukkan m (jumlah baris): 12
Masukkan n (jumlah kolom): 10
Masukkan elemennya,
a00: 0 0 0 0 0 0 1 1 1 13
0 0 0 1 1 1 0 0 15
a01: a02: a03: a04: a05: a06: a07: a08: b0: a10: a11: a12: a13: a14: a15: a16: a17: a18: b1: a20: 1 1 1 0 0 0 0 0 8
a21: a22: a23: a24: a25: a26: a27: a28: b2: a30: 0 0 ,0,04289 0 ,04289 ,0,75 ,0,04289 ,0,75 ,0,61396 14,79
a31: a32: a33: a34: a35: a36: a37: a38: b3: a40: 0 ,0,25 ,0,91421 ,0,25 ,0,91421 ,0,25 ,0,91421 ,0,25 ,0 14,31
a41: a42: a43: a44: a45: a46: a47: a48: b4: a50: 0,61396 ,0,75 ,0,04289 ,0,75 ,0,04289 0 ,0,04289 0 0 ,3,81
0 0 1 0 0 1 0 0 1 18
a51: a52: a53: a54: a55: a56: a57: a58: b5: a60: a61: a62: a63: a64: a65: a66: a67: a68: b6: a70: 0 1 0 0 1 0 0 1 0 12
1 0 0 1 0 0 1 0 6
a71: a72: a73: a74: a75: a76: a77: a78: b7: a80: a81: a82: a83: a84: a85: a86: a87: a88: b8: a90: 0,04289 0,75 ,0,61396 0 ,0,04289 0,75 ,0 0 ,0,04289 10,51
a91: a92: a93: a94: a95: a96: a97: a98: b9: a100: 0,91421 ,0,25 ,0,25 ,0,91421 ,0,25 ,0,25 ,0,91421 ,0,25 ,0,91421 ,16,13
a101: a102: a103: a104: a105: a106: a107: a108: b10: a110: 0,04289 0 0 ,0,75 ,0,04289 0 ,0,61396 ,0,75 ,0,04289 7,04
a111: a112: a113: a114: a115: a116: a117: a118: b11:
Matriks koefisien tidak persegi, tidak bisa menggunakan metode invers.

```

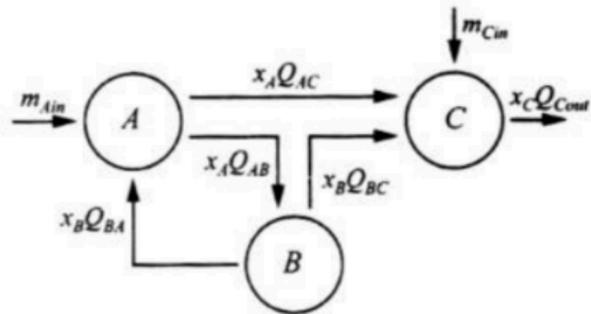
```

SPL

Masukkan m (jumlah baris): 12
Masukkan n (jumlah kolom): 10
Masukkan elemennya,
a00: 0 0 0 0 0 0 1 1 1 13
0 0 0 1 1 1 0 0 15
a01: a02: a03: a04: a05: a06: a07: a08: b0: a10: a11: a12: a13: a14: a15: a16: a17: a18: b1: a20: 1 1 1 0 0 0 0 0 8
a21: a22: a23: a24: a25: a26: a27: a28: b2: a30: 0 0 ,0,04289 0 ,04289 ,0,75 ,0,04289 ,0,75 ,0,61396 14,79
a31: a32: a33: a34: a35: a36: a37: a38: b3: a40: 0 ,0,25 ,0,91421 ,0,25 ,0,91421 ,0,25 ,0,91421 ,0,25 ,0 14,31
a41: a42: a43: a44: a45: a46: a47: a48: b4: a50: 0,61396 ,0,75 ,0,04289 ,0,75 ,0,04289 0 ,0,04289 0 0 ,3,81
0 0 1 0 0 1 0 0 1 18
a51: a52: a53: a54: a55: a56: a57: a58: b5: a60: a61: a62: a63: a64: a65: a66: a67: a68: b6: a70: 0 1 0 0 1 0 0 1 0 12
1 0 0 1 0 0 1 0 6
a71: a72: a73: a74: a75: a76: a77: a78: b7: a80: a81: a82: a83: a84: a85: a86: a87: a88: b8: a90: 0,04289 0,75 ,0,61396 0 ,0,04289 0,75 ,0 0 ,0,04289 10,51
a91: a92: a93: a94: a95: a96: a97: a98: b9: a100: 0,91421 ,0,25 ,0,25 ,0,91421 ,0,25 ,0,25 ,0,91421 ,0,25 ,0,91421 ,16,13
a101: a102: a103: a104: a105: a106: a107: a108: b10: a110: 0,04289 0 0 ,0,75 ,0,04289 0 ,0,61396 ,0,75 ,0,04289 7,04
a111: a112: a113: a114: a115: a116: a117: a118: b11:
Matriks koefisien tidak persegi, tidak bisa menggunakan metode cramer.

```

4) Lihatlah sistem reaktor pada gambar berikut



Dengan laju volume Q dalam m^3/s dan input massa min dalam mg/s . Konservasi massa pada tiap inti reaktor adalah sebagai berikut:

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

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

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

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 $Q_{Cout} = 150 m^3/s$ dan $m_{Ain} = 1300$ dan $m_{Cin} = 200 mg/s$.

Hasil:

SPL

Masukkan m (jumlah baris): 3
Masukkan n (jumlah kolom): 4
Masukkan elemennya.

a00: -120 60 0 -1300
40 -80 0 0
80 20 -150 -200
a01: a02: b0: a10: a11: a12: b1: a20: a21 : a22: b2:

1.000	0.000	0.000	14.4444
0.000	1.000	0.000	7.2222
0.000	0.000	1.000	10.000

Banyak solusi, solusi parametrik:
 $x_1 = 14.44444444444445$
 $x_2 = 7.22222222222222$
 $x_3 = 10.0$

Simpan Hasil ke File?

1. Ya
2. Tidak

Masukkan pilihan: ■

```
... [java - Algeo01-23149] + ⌂ ⌂ ... ^ x

SPL

Masukkan m (jumlah baris): 3
Masukkan n (jumlah kolom): 4
Masukkan elemennya.
a00: -120 60 0 -1300
40 -80 0 0
80 20 -150 -200
a01: a02: b0: a10: a11: a12: b1: a20: a21
: a22: b2:
    1.000   -0.500    0.000   10.833
    0.000    1.000    0.000    7.222
    0.000    0.000    1.000   10.000
Banyak solusi, solusi parametrik:
x1 = 14.444444444444445
x2 = 7.222222222222222
x3 = 10.0

Simpan Hasil ke File?
1. Ya
2. Tidak
Masukkan pilihan: █
```

```
... ☐ java - Algeo01-23149 + ⌂ ⌂ ... ^ x

SPL

Masukkan m (jumlah baris): 3
Masukkan n (jumlah kolom): 4
Masukkan elemennya.
a00: -120 60 0 -1300
40 -80 0 0
80 20 -150 -200
a01: a02: b0: a10: a11: a12: b1: a20: a21: a22
: b2:
Hasil penyelesaian dengan metode Invers adalah
:
x1 = 14.444444444444443
x2 = 7.222222222222221
x3 = 10.000000000000002

Simpan Hasil ke File?
1. Ya
2. Tidak
Masukkan pilihan: █
```

```
... [?] java - Algeo01-23149 + ⚠️ ... ^ x

SPL

Masukkan m (jumlah baris): 3
Masukkan n (jumlah kolom): 4
Masukkan elemennya.
a00: -120 60 0 -1300
40 -80 0 0
80 20 -150 -200
a01: a02: b0: a10: a11: a12: b1: a20: a21: a22
: b2:
Hasil penyelesaian dengan metode Cramer adalah
:
x0 = 14.444444444444445
x1 = 7.222222222222222
x2 = 10.0

Simpan Hasil ke File?
1. Ya
2. Tidak
Masukkan pilihan: █
```

5) Studi Kasus Interpolasi

- 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
--------	-------	-------	-------	-------	-------	-------	-------

Lakukan pengujian pada nilai-nilai berikut:

$$x = 0.2 \quad f(x) = ?$$

$$x = 0.55 \quad f(x) = ?$$

$$x = 0.85 \quad f(x) = ?$$

$$x = 1.28 \quad f(x) = ?$$

Hasil:

```
❖ Masukkan n: 7
Masukkan titik yang diketahui.
x0: 0,1
y0: 0,003
x1: 0,3
y1: 0,067
x2: 0,5
y2: 0,148
x3: 0,7
y3: 0,248
x4: 0,9
y4: 0,370
x5: 1,1
y5: 0,518
x6: 1,3
y6: 0,697
Masukkan absis yang ingin dicari.
x: 0,2
```

Fungsi interpolasi yang memungkinkan adalah:

$$p_6(x) = -0.022976562500000446 + 0.2400000000000079x^1 + 0.1973958333328906x^2 + 1.122873616 \\ 0403815E-13x^3 + 0.02604166666652459x^4 + 8.741380015666164E-14x^5 - 2.0818187548076375E-14x^6$$

Nilai interpolasinya, yakni $p(0.2) = 0.03296093750000065$

Masukkan n: 7

Masukkan titik yang diketahui.

x0: 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

y0: x1: y1: x2: y2: x3: y3: x4: y4: x5: y5: x6: y6: Masukkan absis yang ingin dicari.

x: 0,55

Fungsi interpolasi yang memungkinkan adalah:

$$p_6(x) = -0.022976562500000446 + 0.2400000000000079x^1 + 0.1973958333328906x^2 + 1.122873616 \\ 0403815E-13x^3 + 0.02604166666652459x^4 + 8.741380015666164E-14x^5 - 2.0818187548076375E-14x^6$$

Nilai interpolasinya, yakni $p(0.55) = 0.17111865234375007$

POLINOMIAL INTERPOLATION

Masukkan n: 7

Masukkan titik yang diketahui.

x0: 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

y0: x1: y1: x2: y2: x3: y3: x4: y4: x5: y5: x6: y6: Masukkan absis yang ingin dicari.

x: 0,85

Fungsi interpolasi yang memungkinkan adalah:

$$p_6(x) = -0.022976562500000446 + 0.2400000000000079x^1 + 0.1973958333328906x^2 + 1.122873616 \\ 0403815E-13x^3 + 0.02604166666652459x^4 + 8.741380015666164E-14x^5 - 2.0818187548076375E-14x^6$$

Nilai interpolasinya, yakni $p(0.85) = 0.33723583984375005$

POLINOMIAL INTERPOLATION

Masukkan n: 7

Masukkan titik yang diketahui.

x0: 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

y0: x1: y1: x2: y2: x3: y3: x4: y4: x5: y5: x6: y6: Masukkan absis yang ingin dicari.

x: 1,28

Fungsi interpolasi yang memungkinkan adalah:

$$p_6(x) = -0.022976562500000446 + 0.24000000000000079x^1 + 0.1973958333328906x^2 + 1.122873616 \\ 0403815E-13x^3 + 0.02604166666652459x^4 + 8.741380015666164E-14x^5 - 2.0818187548076375E-14x^6$$

Nilai interpolasinya, yakni $p(1.28) = 0.6775418374999999$

- 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

Tanggal (desimal) adalah tanggal yang sudah diolah ke dalam bentuk desimal 3 angka di belakang koma dengan memanfaatkan perhitungan sebagai berikut:

$$\text{Tanggal (desimal)} = \text{bulan} + (\text{tanggal} / \text{jumlah hari pada bulan tersebut})$$

Sebagai contoh, untuk tanggal 17/06/2022 (dibaca: 17 Juni 2022) diperoleh tanggal(desimal) sebagai berikut:

$$\text{Tanggal (desimal)} = 6 + (17/30) = 6,567$$

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

- 16/07/2022
- 10/08/2022

- 05/09/2022
- Masukan user lainnya berupa tanggal (desimal) yang sudah diolah dengan asumsi prediksi selalu dilakukan untuk tahun 2022.

Hasil:

```
POLINOMIAL INTERPOLATION
```

Masukkan n: 10

Masukkan titik yang diketahui.

x0: 6,567 12624 7 21807 7,258 38391 7,451 54517 7,548 51952 7,839 28228 8,161 35764 8,484 20
813 8,709 12408 9 10534

y0: x1: y1: x2: y2: x3: y3: x4: y4: x5: y5: x6: y6: x7: y7: x8: y8: x9: y9: Masukkan absis y
ang ingin dicari.

x: 7,516

Fungsi interpolasi yang memungkinkan adalah:

$p_9(x) = 7.188430348201641E12 - 9.348572690401229E12x^1 + 5.335014967217705E12x^2 - 1.7570533
415324275E12x^3 + 3.685975674434795E11x^4 - 5.113786485813265E10x^5 + 4.6963169665513315E9x^
6 - 2.755025030644787E8x^7 + 9373741.525538526x^8 - 141006.35265450666x^9$

Nilai interpolasinya, yakni $p(7.516) = 53537.7578125$

```
POLINOMIAL INTERPOLATION
```

Masukkan n: 10

Masukkan titik yang diketahui.

x0: 6,567 12624 7 21807 7,258 38391 7,451 54517 7,548 51952 7,839 28228 8,161 35764 8,484 20
813 8,709 12408 9 10534

y0: x1: y1: x2: y2: x3: y3: x4: y4: x5: y5: x6: y6: x7: y7: x8: y8: x9: y9: Masukkan absis y
ang ingin dicari.

x: 8,323

Fungsi interpolasi yang memungkinkan adalah:

$p_9(x) = 7.188430348201641E12 - 9.348572690401229E12x^1 + 5.335014967217705E12x^2 - 1.7570533
415324275E12x^3 + 3.685975674434795E11x^4 - 5.113786485813265E10x^5 + 4.6963169665513315E9x^
6 - 2.755025030644787E8x^7 + 9373741.525538526x^8 - 141006.35265450666x^9$

Nilai interpolasinya, yakni $p(8.323) = 36294.91015625$

```
POLINOMIAL INTERPOLATION
```

Masukkan n: 10

Masukkan titik yang diketahui.

x0: 6,567 12624 7 21807 7,258 38391 7,451 54517 7,548 51952 7,839 28228 8,161 35764 8,484 20
813 8,709 12408 9 10534

y0: x1: y1: x2: y2: x3: y3: x4: y4: x5: y5: x6: y6: x7: y7: x8: y8: x9: y9: Masukkan absis y
ang ingin dicari.

x: 9,167

Fungsi interpolasi yang memungkinkan adalah:

$p_9(x) = 7.188430348201641E12 - 9.348572690401229E12x^1 + 5.335014967217705E12x^2 - 1.7570533
415324275E12x^3 + 3.685975674434795E11x^4 - 5.113786485813265E10x^5 + 4.6963169665513315E9x^
6 - 2.755025030644787E8x^7 + 9373741.525538526x^8 - 141006.35265450666x^9$

Nilai interpolasinya, yakni $p(9.167) = -667712.2734375$

POLINOMIAL INTERPOLATION

Masukkan n: 10

Masukkan titik yang diketahui.

x0: 6,567 12624 7 21807 7,258 38391 7,451 54517 7,548 51952 7,839 28228 8,161 35764 8,484 20
813 8,709 12408 9 10534

y0: x1: y1: x2: y2: x3: y3: x4: y4: x5: y5: x6: y6: x7: y7: x8: y8: x9: y9: Masukkan absis y
ang ingin dicari.

x: 8,903

Fungsi interpolasi yang memungkinkan adalah:

$p_9(x) = 7.188430348201641E12 - 9.348572690401229E12x^1 + 5.335014967217705E12x^2 - 1.7570533
415324275E12x^3 + 3.685975674434795E11x^4 - 5.113786485813265E10x^5 + 4.6963169665513315E9x^
6 - 2.755025030644787E8x^7 + 9373741.525538526x^8 - 141006.35265450666x^9$

Nilai interpolasinya, yakni $p(8.903) = 50617.046875$

- 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 $h = (2 - 0)/5 = 0.4$.

Hasil: n = 16

```
Masukkan file path: C:\ITB\Belajar\IF\Sem 3\IF2123 Aljabar Linier dan Geometri\Tubes\Tubes 1\test\interpolasiPolinomTest.txt
Fungsi interpolasi yang memungkinkan adalah:
p16(x) = 0.0 + 5.536021587006992x^1 - 48.78275475096094x^2 + 297.0505333120965x^3 - 1252.115937708222x^4 + 3790.406903331744x^5 - 8478.596727144346x^6 + 16290.966004727707x^7 - 18
369.209379316202x^8 + 18099.492829562103x^9 - 13651.497295495225x^10 + 7813.317193581992x^11 - 3332.210500080145x^12 + 1025.26005864881x^13 - 214.9274561899471x^14 + 27.461315598
126532x^15 - 1.6129248280067383x^16
Nilai interpolasinya, yakni p(1.0) = 0.537882842739366

Simpan Hasil ke File?
1. Ya
2. Tidak
```

6) Studi Kasus Regresi Linear dan Kuadratik Berganda

Diberikan sekumpulan data sesuai pada tabel berikut ini.

Table 12.1: Data for Example 12.1

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

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

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

Dari data-data tersebut, apabila diterapkan *Normal Estimation Equation for Multiple Linear Regression*, maka diperoleh sistem persamaan linear sebagai berikut.

$$\begin{aligned} 20b_0 + 863.1b_1 + 1530.4b_2 + 587.84b_3 &= 19.42 \\ 863.1b_0 + 54876.89b_1 + 67000.09b_2 + 25283.395b_3 &= 779.477 \\ 1530.4b_0 + 67000.09b_1 + 117912.32b_2 + 44976.867b_3 &= 1483.437 \\ 587.84b_0 + 25283.395b_1 + 44976.867b_2 + 17278.5086b_3 &= 571.1219 \end{aligned}$$

Silahkan terapkan model-model ini pada *Multiple Quadratic Equation* juga dan bandingkan hasilnya. Sistem persamaan linear tidak akan diberikan untuk kasus ini.

Hasil: Regresi Linear Berganda

Enter any input to continue..

1,000	72,400	76,300	29,180	0,900
1,000	41,600	70,300	29,350	0,910
1,000	34,300	77,100	29,240	0,960
1,000	35,100	68,000	29,270	0,890
1,000	10,700	79,000	29,780	1,000
1,000	12,900	67,400	29,390	1,100
1,000	8,300	66,800	29,690	1,150
1,000	20,100	76,900	29,480	1,030
1,000	72,200	77,700	29,090	0,770
1,000	24,000	67,700	29,600	1,070
1,000	23,200	76,800	29,380	1,070
1,000	47,400	86,600	29,350	0,940
1,000	31,500	76,900	29,630	1,100
1,000	10,600	86,300	29,560	1,100
1,000	11,200	86,000	29,480	1,100
1,000	73,300	76,300	29,400	0,910
1,000	75,400	77,900	29,280	0,870
1,000	96,600	78,700	29,290	0,780
1,000	107,400	86,800	29,030	0,820
1,000	54,900	70,900	29,370	0,950

Fungsi regresi linear berganda yang memungkinkan adalah:
 $p(x_1, x_2, x_3) = -3.507778140883147 - 0.0026249907458783875x_1 + 7.989410472218425E-4x_2 + 0.15415503019828913x_3$

Nilai regresi linear bergandanya adalah:
 $p(50.0, 76.0, 29.3) = 0.9384342262216654$

Simpan Hasil ke File?

1. Ya
2. Tidak

Masukkan pilihan:

Regresi Kuadratik Berganda

```
Masukkan file path: test/regresiTest.txt
Enter any input to continue..

1,000 72,400 76,300 29,1805241,7605821,690 851,4725524,1202226,4342112,632 0,900
1,000 41,600 70,300 29,3501730,5604942,090 861,4232924,4802063,3051220,960 0,910
1,000 34,300 77,100 29,2401176,490594,410 854,978264,5302254,4041002,932 0,960
1,000 35,100 68,000 29,2701232,0104624,000 856,7332386,8001990,3601027,377 0,890
1,000 10,700 79,000 29,780 114,4906241,000 886,844 845,3002352,620 318,646 1,000
1,000 12,900 67,400 29,390 166,4104542,760 863,772 869,4601980,886 379,131 1,100
1,000 8,300 66,800 29,690 68,8904462,240 881,496 554,4401983,292 246,427 1,150
1,000 20,100 76,900 29,488 404,0105913,610 869,0701545,6902267,012 592,548 1,030
1,000 72,200 77,700 29,6905212,8406037,290 846,2285697,9402260,2932100,298 0,770
1,000 24,000 67,700 29,600 576,0004583,290 876,1601624,8002003,220 710,400 1,070
1,000 23,200 76,800 29,380 538,2405898,240 863,1841781,7602256,384 681,416 1,070
1,000 47,400 86,600 29,3502246,7607499,560 861,4234104,8402541,7101391,190 0,940
1,000 31,500 76,900 29,630 992,2805913,610 877,9372422,3502278,547 933,345 1,100
1,000 10,600 86,300 29,560 112,3607447,691 873,794 914,7802551,028 313,336 1,100
1,000 11,200 86,000 29,488 125,4407396,000 869,074 963,2002535,280 330,176 1,100
1,000 73,300 76,300 29,4005372,8905821,690 864,3605592,7902243,2202155,020 0,910
1,000 75,400 77,900 29,2805685,1606068,410 857,3138573,6602280,9122207,712 0,870
1,000 96,600 78,700 29,2909331,5606193,690 857,9847602,4202305,1232829,414 0,780
1,000 107,400 86,800 29,03011534,7607534,240 842,7419322,3202519,8043117,822 0,820
1,000 54,900 70,900 29,3703014,0105026,810 862,5973892,4102082,3331612,413 0,950

Fungsi regresi kuadratik berganda yang memungkinkan adalah:
p(x1, x2, x3) = -1146.4372177586572 + 0.18385564577647335x1 + 75.45028048207061x3 - 1.483572711875502E-6x4 - 2.3765001108417427E-4x5 - 1.2403842319143972x6 +
1.700392172103338E-5x7 - 0.027248118399374112x8 - 0.00641282228494654x9

Dengan definisi:
x1 = x1
x2 = x2
x3 = x3
x4 = x1^2
x5 = x2^2
x6 = x3^2
x7 = x1*x2
x8 = x2*x3
x9 = x3*x1

Nilai regresi kuadratik bergandanya adalah:
p(50,0, 76,0, 29,3) = 0.9439970941991012
```

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.

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

Tentukan nilai:

$$\begin{aligned} f(0, 0) &= 21 \\ f(0.5, 0.5) &= 87.796 \\ f(0.25, 0.75) &= 117.732 \\ f(0.1, 0.9) &= 128.575 \end{aligned}$$

Hasil:

```
... [x] java - Algeo01-23149 + ⚖ ... ^ x

BICUBIC SPLINE INTERPOLATION

Masukkan matriks:
21 98 125 153
51 101 161 59
0 42 72 210
16 12 81 96
0 0
Nilai interpolasi pada (0.0, 0.0) adalah: 21.0

Simpan Hasil ke File?
1. Ya
2. Tidak
Masukkan pilihan: 1
```

```
... [x] java - Algeo01-23149 + ⌂ ⌂ ⌂ ... ^ x

BICUBIC SPLINE INTERPOLATION

Masukkan matriks:
21 98 125 153
51 101 161 59
0 42 72 210
16 12 81 96
0.5 0.5
Nilai interpolasi pada (0.5, 0.5) adalah: 87.796875

Simpan Hasil ke File?
1. Ya
2. Tidak
Masukkan pilihan: 1
```

```
... [x] java - Algeo01-23149 + ⌂ ⌂ ... ⌈ ⌉
BICUBIC SPLINE INTERPOLATION

Masukkan matriks:
21 98 125 153
51 101 161 59
0 42 72 210
16 12 81 96
0.25 0.75
Nilai interpolasi pada (0.25, 0.75) adalah: 117.732
177734375

Simpan Hasil ke File?
1. Ya
2. Tidak
Masukkan pilihan: 1
```

```
... [ ] java - Algeo01-23149 + ⌂ ⌂ ... ⌂ x

BICUBIC SPLINE INTERPOLATION

Masukkan matriks:
21 98 125 153
51 101 161 59
0 42 72 210
16 12 81 96
0.1 0.9
Nilai interpolasi pada (0.1, 0.9) adalah: 128.57518
700000003

Simpan Hasil ke File?
1. Ya
2. Tidak
Masukkan pilihan: [ ]
```

8) Studi Kasus untuk Image Resizing dan Stretching

Masukan berupa suatu image dengan format jpg dan scale width dan height baru yang diinginkan.

test >  test_image.jpg



Dari *image* di atas kemudian dilakukan suatu image resizing dengan menggunakan metode bicubic spline interpolation. Pada contoh ini digunakan scale width 1.5 dan scale height 2. Hasilnya sebagai berikut

test >  new_image.jpg



BAB V

Kesimpulan

Kesimpulan

Kami telah berhasil membuat pustaka dalam Bahasa Java. Pustaka tersebut dapat menemukan solusi SPL dengan metode eliminasi Gauss, metode eliminasi Gauss-Jordan, metode matriks balikan, dan kaidah *Cramer*. Pustaka tersebut juga dapat menghitung determinan matriks dengan Reduksi Baris dan Ekspansi Kofaktor. Selain itu, pustaka tersebut juga dapat menghitung matriks balikan dengan metode matriks adjoin dan matriks identitas, menghitung interpolasi polinom, interpolasi bicubic spline, regresi linier berganda, dan image resizing dan stretching.

Saran

- Seringkali ada hasil yang tidak sesuai (misal: seharusnya keluar 1, namun yang keluar adalah 0.99999999999999). Hal ini dapat terjadi karena penggunaan double. Untuk kedepannya, lebih baik menggunakan Bigdecimal.
- Parametrik harus di-*handle* dengan hati-hati bila ingin dibuat kembali kedepannya.
- Perhatikan kembali kasus khusus bila baris lebih dari kolom.

Refleksi

- Jangan *deadliner!*
- Harus lebih banyak memperhatikan *edge cases*. Kami sempat tersendat pengumpulan akibat ada *edge cases* berupa baris yang lebih panjang dari kolom.

LAMPIRAN

1. Asistensi 1 (14 Oktober 2024)

Hasil:

1. GUI apakah diperlukan? JAWAB: Kalau bisa pakai lebih baik
2. Pada kasus perhitungan ada ketidakpresisionan, seperti angka 1 yang direpresentasikan sebagai 0,999... . JAWAB: Tidak apa, lakukan pembulatan kalau bisa

2. Referensi

<https://informatika.stei.itb.ac.id/~rinaldi.munir/AljabarGeometri/2023-2024/Algeo-03-Sistem-Persamaan-Linier-2023.pdf>

<https://informatika.stei.itb.ac.id/~rinaldi.munir/AljabarGeometri/2023-2024/Algeo-04-Tiga-Kemungkinan-Solusi-SPL-2023.pdf>

<https://informatika.stei.itb.ac.id/~rinaldi.munir/AljabarGeometri/2023-2024/Algeo-05-Sistem-Persamaan-Linier-2-2023.pdf>

3. Tautan Repository

<https://github.com/reletz/Algeo01-23149/activity>