Laporan Tugas Besar
IF2123 Aljabar Linear dan Geometri
Semester I Tahun 2024/2025

Sistem Persamaan Linier, Determinan, dan Aplikasinya

Kelompok xx (K01):
Faqih Muhammad Syuhada (13523057)
Nayla Zahira (13523079)
Azfa Radhiyya Hakim (13523115)

SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA
INSTITUT TEKNOLOGI BANDUNG
2024

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB I
DESKRIPSI MASALAH

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

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

1.1. Spesifikasi

1. Program dapat menerima masukan (input) baik dari keyboard maupun membaca masukan dari file text. Untuk SPL, masukan dari keyboard adalah m, n, koefisien aij , dan bi. 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 aij. 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 adjoin.

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, (x0, y0), (x1, y1), …, (xn, yn), 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}, ..., x 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

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

dan untuk regresi adalah

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

untuk kasus regresi kuadratik, variabel boleh menggunakan x1, x2, 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 x 4 yang berisi konfigurasi nilai fungsi dan turunan berarah disekitarnya, diikuti dengan nilai a dan b untuk mencari nilai f(a, b). Misalnya jika nilai dari $f(0, 0)$, $f(1, 0)$, $f(0, 1)$, $f(1, 1)$, $f_x(0, 0)$, $f_x(1, 0)$, $f_x(0, 1)$, $f_x(1, 1)$, $f_y(0, 0)$, $f_y(1, 0)$, $f_y(0, 1)$, $f_y(1, 1)$, $f_{xy}(0, 0)$, $f_{xy}(1, 0)$, $f_{xy}(0, 1)$, $f_{xy}(1, 1)$ 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 Persamaaan Linier
 2. Determinan
 3. Matriks balikan
 4. Interpolasi Polinom
 5. Interpolasi Bicubic Spline
 6. Regresi linier dan kuadratik berganda
```

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

2.1. Determinan

Dalam konteks matriks, determinan adalah suatu fungsi yang menghubungkan sebuah bilangan riil ke suatu matriks persegi. Matriks persegi yang dimaksud adalah suatu matriks yang memiliki jumlah baris dan kolom yang sama. Determinan dari suatu matriks A yang berukuran nxn dapat dinotasikan sebagai det(A) atau |A| dan dapat diilustrasikan sebagai berikut.

$$|A| = \det(A) = \begin{vmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \dots & \dots & \dots & \dots \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{vmatrix}$$

Gambar 2.1. Notasi Determinan

Nilai determinan dari suatu matriks didapat melalui operasi unsur-unsur matriks tersebut. Terdapat dua metode utama dalam menghitung determinan matriks, yaitu metode ekspansi kofaktor dan reduksi baris.

2.1.1. Determinan dengan Ekspansi Kofaktor

Metode ekspansi kofaktor didasarkan pada konsep kofaktor dan aturan ekspansi kofaktor. Dalam metode ini, nilai determinan dapat dihitung dengan menjumlahkan perkalian dari setiap elemen a_{ij} dengan kofaktor entri a_{ij} pada suatu baris atau kolom.

$$A = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{bmatrix}$$

- Didefinisikan:

  M_{ij} = minor entri a_{ij}

  = determinan upa-matriks (*submatrix*) yang elemen-elemennya tidak berada pada baris i dan kolom j

  $C_{ij} = (-1)^{i+j} M_{ij}$ = kofaktor entri a_{ij}

Gambar 2.2. Definisi kofaktor entri a

$$
\begin{aligned}
\det(A) &= a_{11}C_{11} + a_{12}C_{12} + \dots + a_{1n}C_{1n} \\
\det(A) &= a_{21}C_{21} + a_{22}C_{22} + \dots + a_{2n}C_{2n} \\
&\vdots \\
\det(A) &= a_{n1}C_{n1} + a_{n2}C_{n2} + \dots + a_{nn}C_{nn}
\end{aligned}
$$

Secara baris

$$
\begin{aligned}
\det(A) &= a_{11}C_{11} + a_{21}C_{21} + \dots + a_{n1}C_{n1} \\
\det(A) &= a_{12}C_{12} + a_{22}C_{22} + \dots + a_{n2}C_{n2} \\
&\vdots \\
\det(A) &= a_{1n}C_{1n} + a_{2n}C_{2n} + \dots + a_{nn}C_{nn}
\end{aligned}
$$

Secara kolom

Gambar 2.3. Perhitungan dengan ekspansi kofaktor secara baris dan kolom

2.1.2. Determinan dengan Reduksi Baris

Metode reduksi baris adalah salah satu metode yang dapat digunakan untuk menghitung determinan dari sebuah matriks persegi. Metode ini menggunakan operasi baris elementer(OBE) untuk mengubah bentuk matriks menjadi bentuk matriks segitiga atas atau matriks segitiga bawah.

$$
A = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ 0 & a_{22} & a_{23} & a_{24} \\ 0 & 0 & a_{33} & a_{34} \\ 0 & 0 & 0 & a_{44} \end{bmatrix}
$$

Gambar 2.4. Matriks segitiga atas

$$
A = \begin{bmatrix} a_{11} & 0 & 0 & 0 \\ a_{21} & a_{22} & 0 & 0 \\ a_{31} & a_{32} & a_{33} & 0 \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix}
$$

Gambar 2.5. Matriks segitiga bawah

Ketika matriks berbentuk segitiga atas atau segitiga bawah, determinan dapat dihitung dengan mengalikan elemen-elemen pada diagonal utama. Secara umum, untuk matriks segitiga nxn, berlaku:

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

Dengan p merupakan banyaknya pertukaran baris yang dilakukan dalam OBE.

2.2. Matriks Kofaktor dan Matriks Adjoin

2.3. Matriks Balikan

Matriks balikan (inverse matrix) dari suatu matriks persegi A berukuran n x n didefinisikan sebagai matriks A^{-1}, yang memenuhi syarat $AA^{-1} = A^{-1}A = I$, di mana I adalah matriks identitas berukuran n x n. Matriks identitas ini memiliki elemen 1 pada diagonal utamanya dan elemen 0 di tempat lainnya. Suatu matriks A memiliki matriks balikan jika dan hanya jika determinannya tidak sama dengan nol. Terdapat dua metode utama untuk menghitung matriks balikan yaitu dengan metode adjoin dan metode eliminasi Gauss-Jordan (OBE)

2.3.1. Matriks Balikan dengan Adjoin

Metode adjoin memanfaatkan konsep kofaktor dan determinan untuk mendapatkan matriks balikan dari suatu matriks. Matriks balikan ini dapat dihitung sesuai dengan rumus berikut.

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

2.3.2. Matriks Balikan dengan Eliminasi Gauss-Jordan

Matriks balikan juga dapat dihitung dengan menggunakan metode eliminasi Gauss-Jordan. Untuk matriks A yang berukuran nxn, matriks balikannya, dapat dicari dengan cara berikut:

$$[A|I] \overset{\text{G-J}}{\sim} [I|A^{-1}]$$

Gambar 2.6. Menghitung Matriks Balikan dengan Eliminasi Gauss Jordan

2.4. Sistem Persamaan Linear

2.4.1. Metode Eliminasi Gauss

Metode eliminasi Gauss adalah suatu teknik untuk memperoleh penyelesaian dari sebuah sistem persamaan linier dengan memanipulasi bentuk *augmented matrix* menjadi bentuk yang lebih sederhana, yaitu bentuk matriks eselon baris (*row echelon form)*. Sebuah matriks dikatakan dalam bentuk eselon baris jika memenuhi syarat-syarat berikut:

1. Jika sebuah baris tidak seluruhnya terdiri dari nol, angka pertama yang bukan nol harus 1 (“leading one”).
2. Jika terdapat baris yang seluruh elemennya nol, baris tersebut harus diletakkan di baris bawah matriks.
3. Setiap “leading one” di baris lebih bawah berada di kolom yang lebih kanan daripada “leading one” di baris atasnya.

$$
\begin{bmatrix} 1 & * & * & * \\ 0 & 0 & 1 & * \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}
\qquad
\begin{bmatrix} 0 & 1 & * & * & * \\ 0 & 0 & 0 & 1 & * \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}
$$

Keterangan : * adalah sembarang nilai

Gambar 2.6. Matriks Eselon Baris

Proses pembentukan matriks eselon baris dilakukan melalui operasi baris elementer (OBE) yang terdiri dari tiga jenis operasi.

1. Perkalian satu baris dengan suatu bilangan bukan nol.
2. Pertukaran dua baris.
3. Penambahan kelipatan dari satu baris ke baris lainnya.

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

Gambar 2.7. Metode Eliminasi Gauss

Setelah penyederhanaan matriks ke bentuk eselon baris, akan dihasilkan sejumlah persamaan yang lebih mudah untuk diselesaikan. Sehingga, solusi sistem persamaan dapat ditemukan cukup dengan substitusi mundur.

2.4.2. Metode Eliminasi Gauss Jordan

Metode Eliminasi Gauss-Jordan adalah pengembangan dari metode eliminasi Gauss yang digunakan untuk menyelesaikan sistem persamaan linear. Metode ini mengubah matriks augmented menjadi bentuk eselon baris tereduksi (reduced row echelon form), sehingga solusi dari sistem persamaan bisa langsung diperoleh tanpa perlu substitusi mundur.

Adapun ciri-ciri matriks eselon tereduksi adalah sebagai berikut:

1. Jika sebuah baris tidak seluruhnya terdiri dari nol, angka pertama yang bukan nol harus 1 (“leading one”).
2. Jika terdapat baris yang seluruh elemennya nol, baris tersebut harus diletakkan di baris bawah matriks.
3. “Leading one” dalam suatu baris berada di sebelah kanan leading one pada baris di atasnya.
4. Setiap “leading one” adalah satu-satunya angka bukan nol di kolomnya.

$$\begin{bmatrix} 1 & * & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad \text{atau} \quad \begin{bmatrix} 0 & 1 & 0 & 0 & * \\ 0 & 0 & 1 & 0 & * \\ 0 & 0 & 0 & 1 & * \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Gambar 2.8. Matriks Eselon Baris Tereduksi

2.4.3. Metode Matriks Balikan

2.4.4. Kaidah Cramer

Kaidah Cramer juga menjadi salah satu alternatif metode dalam menyelesaikan sistem persamaan linear dengan jumlah banyaknya persamaan sama dengan banyaknya variabel.. Penyelesaian sistem persamaan linear menggunakan Kaidah Cramer adalah dengan mencari nilai determinan dari setiap matriks koefisien A dengan konstanta b pada Ax=B. Untuk setiap variabel dalam sistem persamaan linear, Kaidah Cramer menghitung solusinya secara terpisah.

Dengan metodenya ini, kaidah cramer memiliki beberapa pembatasan penting. Perama, sistem persamaan linear yang ingin dicari solusinya harus matriks persegi. Kedua, determinan matriks A harus tidak nol, dimana jika sistem persamaan linear dengan determinannya nol, maka sistem persamaan linear tidak memiliki solusi unik. Sistem persamaan linear yang besar juga tidak efisien jika diselesaikan menggunakan Kaidah Cramer. Hal tersebut terjadi karena menghitung determinan matriks merupakan tugas yang kompleks dan memakan waktu.

$$x_1 = \frac{\det(A_1)}{\det(A)}, \quad x_2 = \frac{\det(A_2)}{\det(A)}, \quad \ldots, \quad x_n = \frac{\det(A_n)}{\det(A)}$$

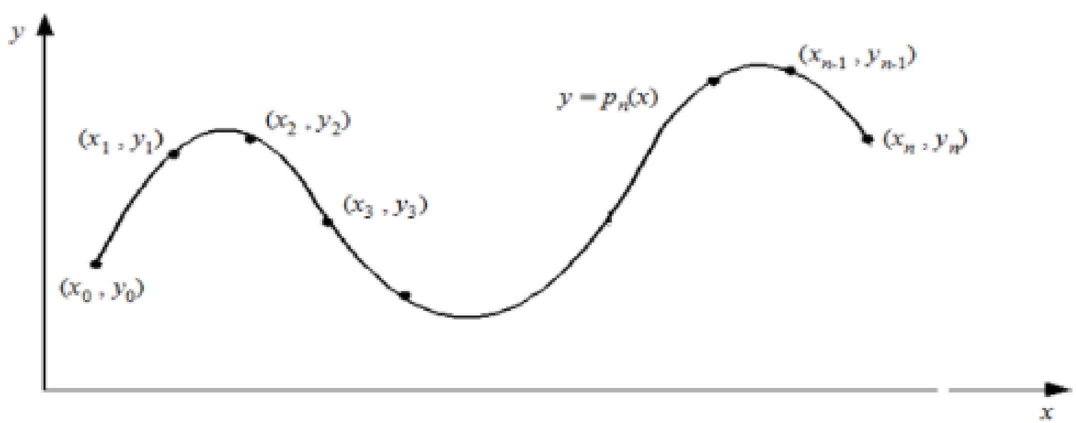
Gambar 2.9. Kaidah Cramer

Metode penyelesaian sistem persamaan linear Ax=B yang terdiri dari n persamaan linear menggunakan Kaidah Cramer dengan peubah sedemikian sehingga det(A) ≠ 0. Maka, sistem persamaan linear tersebut memiliki solusi yang unik dengan metode substitusi seperti pada Gambar 2.10.

2.5. Interpolasi

2.5.1. Interpolasi Polinomial

Interpolasi Polinom adalah suatu metode matematis yang digunakan untuk menemukan polinom yang melewati sejumlah titik berbeda atau data yang diketahui yaitu polinom Pn(X), dengan Pn(Xi)=Yi untuk setiap data (xi,yi). Polinom ini juga dapat digunakan untuk memperkirakan nilai y di sembarang titik dalam selang [x0,xn]. Berikut ilustrasi beberapa titik yang diinterpolasi secara polinomial.



Gambar 2.10. Interpolasi Polinomial

Polinom Interpolasi derajat n yang menginterpolasi titik-titik (x0,y0), (xi,yi), ….(xn,yn), berbentuk Pn(X)=a0…. Jika hanya ada dua titik, (x0, y0) dan (x1, y1), maka polinom yang menginterpolasi kedua titik tersebut adalah p1(x) =

a0 + a1x yaitu berupa persamaan garis lurus. Jika tersedia tiga titik, (x0, y0), (x1, y1), dan (x2, y2), maka polinom yang menginterpolasi ketiga titik
tersebut adalah p2(x) = a0 + a1x + a2x2 atau persaman kuadrat dan kurvanya berupa parabola. Jika tersedia empat titik, (x0, y0), (x1, y1), (x2,y2), dan (x3, y3), polinom yang menginterpolasi keempat titik tersebut adalah p3(x) = a0 + a1x + a2x2+a3x3, demikian seterusnya.

2.5.2. Interpolasi *Bicubic Spline*

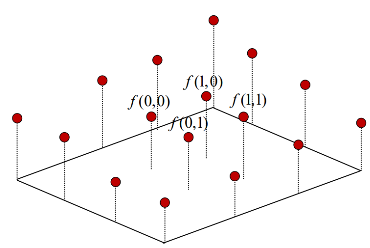
Interpolasi *Bicubic Spline* adalah metode interpolasi yang digunakan untuk mengaproksimasi nilai sebuah fungsi di antara titik-titik data yang diketahui. Metode ini memperluas konsep interpolasi kubik satu dimensi ke dalam dua dimensi, di mana fungsi polinomial kubik digunakan untuk menghubungkan titik-titik dalam grid data, baik secara horizontal maupun vertikal. *Bicubic spline interpolation* ini melibatkan konsep spline dan konstruksi serangkaian polinomial kubik di dalam setiap sel segi empat dari data yang ada. Pendekatan ini menciptakan permukaan yang halus dan kontinu, memungkinkan untuk perluasan data secara visual yang lebih akurat daripada metode interpolasi linear.

Dalam interpolasi ini, 16 titik data digunakan, terdiri dari 4 titik referensi utama di pusat dan 12 titik tambahan di sekitarnya, yang berfungsi untuk menghitung turunan parsial dari titik-titik pusat tersebut. 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 2.11. Ilustrasi Pemodelan *Bicubic Spline Interpolation*

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

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

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

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

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

Gambar 2.12. Persamaan polinomial dalam pemodelan bicubic spline

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

$$y = Xa$$

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

Gambar 2.13. Matriks Hasil Persamaan Polinomial

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.

2.6. Regresi Berganda

2.6.1. Regresi Linear Berganda

Regresi Linear Berganda adalah suatu metode statistik yang digunakan untuk menganalisis hubungan antara variabel dependen dengan dua atau lebih variabel independen. Meskipun sudah ada persamaan jadi untuk menghitung regresi linear sederhana, terdapat persamaan umum dari regresi linear yang biasa digunakan untuk regresi linear berganda.

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

Gambar 2.14. Persamaan umum regresi linear

Untuk mendapat nilai dari setiap β_i dapat digunakan NEE (*Normal Estimation Equation*) sebagai berikut

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

Gambar 2.15. *Normal Estimation Equationfor Multiple Linear Regression*

2.6.2. Regresi Kuadratik Berganda

Regresi berganda terdiri atas 3 variabel utama, yaitu variabel linear, variabel kuadrat, dan variabel interaksi. Untuk setiap n peubah, jumlah variabel linear, kuadrat, dan interaksi akan berbeda-beda. Berikut adalah contoh regresi kuadratik 2 variabel peubah.

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

Gambar 2.16. *Contoh Regresi Kuadratik 2 Variabel Peubah*

N menandakan jumlah peubah, terdapat 2 variabel linier yaitu ui dan vi, 2 variabel kuadrat yaitu ui2 dan vi2, 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 C2n variabel linier (dengan syarat n > 1).

BAB III
IMPLEMENTASI PUSTAKA DAN PROGRAM

3.1. Struktur Kode Program

gambar

3.2. Struktur Kelas (Atribut dan Metode)

3.2.1. Folder matrix

3.2.1.1. Matrix.java

Tabel 3.1. Atribut dari *Class* Matriks

Nama Atribut	Keterangan
private int row	Mendeklarasikan jumlah baris matriks dengan tipe integer
private int col	Mendeklarasikan jumlah kolom matriks dengan tipe integer
private double[][] matrix	Mendeklarasikan array of array yang merupakan inisialisasi matriks

Tabel 3.2. Metode dari *Class* Matriks

Nama Metode	Keterangan
public Matrix(int row, int col)	Fungsi untuk membentuk matriks dengan jumlah baris = row dan jumlah kolom = col.
public int getRowEff()	Fungsi untuk mendapatkan jumlah baris dari suatu matriks.
public int getColEff()	Fungsi untuk mendapatkan jumlah kolom

	dari suatu matriks.
public int getLastRowIdx()	Fungsi untuk mendapatkan indeks baris terakhir.
public int getLastColIdx()	Fungsi untuk mendapatkan indeks dari kolom terakhir.
public boolean isSquare()	Mengecek apabila matriks berukuran nxn atau memiliki jumlah baris dan kolom yang sama.
public double getElmt(int i, int j)	Fungsi untuk mendapatkan elemen matriks[i][j]
public void setElmt(int i, int j, double elmt)	Prosedur untuk menetapkan elemen elmt pada matrix[i][j].
public Matrix getRowElmnt(int row)	Fungsi yang membentuk matriks baru berisi baris row.
public Matrix getColElmnt(int col)	Fungsi yang membentuk matriks baru berisi baris row.
public void rowSwap(Matrix m, int rows1, int rows2)	Prosedur untuk menukar baris row1 dan baris row2 pada matriks M.
public Matrix copyMatrix()	Fungsi untuk mendapatkan salinan dari matriks yang dipanggil.
public Matrix transposeMatrix()	Fungsi untuk mendapatkan bentuk transpos dari suatu matriks.
public static Matrix subMatrix(Matrix m, int rowCoret,	Fungsi yang menghasilkan submatriks dari matriks m. Submatriks ini merupakan

int colCoret)	matriks m *exclude* baris rowCoret dan kolom colCoret.
public Matrix getColumn(int colIndex)	Fungsi yang mengeluarkan kolom ke collIndex dari matriks.
public void displayMatrix()	Prosedur untuk menampilkan matriks ke terminal.

3.2.1.2. InputOutput.java

Tabel 3.3. Atribut dari Class InputOutput

Nama Atribut	Keterangan
private static Scanner scanInterpol = new Scanner(System.in)	Membuat scanner untuk menginput persoalan interpolasi polinomial

Tabel 3.4. Metode dari *Class* InputOutput

Nama Metode	Keterangan
public static Matrix readMatrixFromKeyboard()	Fungsi untuk membaca input matriks dari keyboard.
public static Matrix readMatrixSquareFromKeyboard()	Fungsi untuk membaca input matriks persegi dari keyboard.
public static Matrix readMatrixFromFile(String path)	Fungsi untuk membaca input matriks dari suatu file.
public static Matrix readRegresiKeyboard()	Fungsi untuk membaca input matriks untuk persoalan regresi.
public static Matrix readInterpolasi()	Fungsi untuk membaca input matriks untuk persoalan interpolasi.

public static Object[] bacaFileKeMatriks(String filePath)	Fungsi untuk membaca input matriks untuk persoalan bicubic.
public static void printMatrix(Matrix m)	Prosedur untuk menampilkan matriks m ke terminal
public static String matrixToString(Matrix M)	Fungsi untuk meng-*convert* suatu matrix M ke string untuk nantinya dibuat file
public static void saveOutputToFile(String output)	Prosedur untuk menyimpan suatu string output ke file.
public static int lineCount(String path)	Fungsi untuk membaca jumlah *line* suatu file
public static int columnCount(String path)	Fungsi untuk membaca jumlah kolom suatu file

3.2.2. Folder spl

3.2.2.1. Determinan.java

Tabel 3.5. Atribut dari *Class* Determinan

Nama Atribut	Keterangan
-	Tidak terdapat atribut pada kelas ini

Tabel 3.6. Metode dari *Class* Determinan

Nama Metode	Keterangan
public static double detOBE(Matrix m)	Fungsi untuk menghitung determinan matrix m dengan menggunakan metode reduksi baris

public static double detKofaktor(Matrix M)	Fungsi untuk menghitung determinan matrix m dengan menggunakan metode ekspansi kofaktor secara rekursif

3.2.2.2. BalikanMatriks.java

Tabel 3.7. Atribut dari *Class* BalikanMatriks

Nama Atribut	Keterangan
private static final double EPSILON	Mendeklarasikan nilai EPSILON untuk digunakan sebagai toleransi ketelitian pada perhitungan bertipe double

Tabel 3.8. Metode dari *Class* BalikanMatriks

Nama Metode	Keterangan
public static Matrix balikanMatrixOBESolution(Matrix M)	Fungsi untuk menghasilkan matriks balikan dari suatu matriks M menggunakan metode OBE
public static double getCofactor(Matrix M, int i, int j)	Fungsi yang menghasilkan elemen untuk matriks kofaktor M[i][j]
public static Matrix mAdjoin(Matrix M)	Fungsi untuk menghasilkan matriks adjoin dari matrix M
public static Matrix balikanMatrixKofaktorSolution(Matrix M)	Fungsi untuk menghasilkan matriks balikan dari suatu matriks M menggunakan metode kofaktor

3.2.2.3. Gauss.java

Tabel 3.9. Atribut dari *Class* Gauss

Nama Atribut	Keterangan
private static final double EPSILON	Mendeklarasikan nilai EPSILON untuk digunakan sebagai toleransi ketelitian pada perhitungan bertipe double

Tabel 3.10. Metode dari *Class* Gauss

Nama Metode	Keterangan
public Matrix multiplyRow(Matrix M, double coef, int row)	Fungsi untuk mengalikan sebuah baris pada matrix M dengan sebuah konstanta coef dan mengeluarkan matriks M yang baru.
public void plusRow(Matrix M, int targetRow, int sourceRow, double coef)	Prosedur yang menambah baris targetRow dengan coef kali baris sourceRow. Fungsi mengembalikan Matriks M yang baru.
public static void sortZero(Matrix M, int col)	Prosedur untuk menukar baris yang memiliki nilai 0 pada kolom col dengan baris yang memiliki nilai bukan nol.
public static Matrix gaussSolution(Matrix M)	Fungsi untuk menghasilkan matriks dengan bentuk eselon baris (eliminasi gauss)
public static Matrix printSolutionGauss(Matrix M)	Fungsi untuk menampilkan hasil penyelesaian spl menggunakan metode eliminasi gauss ke terminal.

3.2.2.4. GaussJordan.java

Tabel 3.11. Atribut dari *Class* GaussJordan

Nama Atribut	Keterangan
private static final double EPSILON	Mendeklarasikan nilai EPSILON untuk digunakan sebagai toleransi ketelitian pada perhitungan bertipe double

Tabel 3.12. Metode dari *Class* GaussJordan

Nama Metode	Keterangan
public static Matrix gaussJordanSolution(Matrix M)	Fungsi untuk menghasilkan matriks dengan bentuk eselon baris tereduksi (eliminasi gauss jordan)
public static Matrix printSolutionGaussJordan(Matrix M)	Fungsi untuk menampilkan hasil penyelesaian spl menggunakan metode gauss jordan ke terminal.

3.2.2.5. Cramer.java

Tabel 3.13. Atribut dari *Class* Cramer

Nama Atribut	Keterangan
-	Tidak terdapat atribut pada kelas ini

Tabel 3.14. Metode dari *Class* Cramer

Nama Metode	Keterangan
public static Matrix cramerSolution (Matrix m)	Fungsi untuk menghasilkan matriks solusi sistem persamaan linear dengan menggunakan kaidah cramer.

3.2.3. Folder interpolasi

3.2.3.1. InterPolim.java

Tabel 3.15. Atribut dari *Class* interPolim

Nama Atribut	Keterangan
private static Scanner scanInterpol = new Scanner(System.in)	Membuat scanner untuk menginput persoalan interpolasi polinomial

Tabel 3.16. Metode dari *Class* interPolim

Nama Metode	Keterangan
public static void interPolinom(Matrix A)	Prosedur ini digunakan untuk mendapatkan fungsi interpolasi polinom dan hasil taksiran x. Inputan titik-titiknya berasal dari ketikan keyboard
public static void interPolinomFile(Matrix A, String path)	Prosedur ini digunakan untuk mencari fungsi interpolasi polinom dan hasil taksiran x. Inputan titik-titiknyaberasal dariketikankeyboard
private static double calculateEstimation(Matrix result, double x)	Fungsi ini digunakan untuk menghitung nilai estimasi polinomial pada titik x tertentu.
private static Matrix SPLInterpol(Matrix A)	Fungsi ini membangun matriks koefisien polinomial dari titik-titik dalam matriks A dan menyelesaikan SPL menggunakan eliminasi Gauss-Jordan.

Nama Metode	Keterangan
private static int countTest(String pathToFile)	Fungsi ini digunakan untuk menghitung berapa banyak titik uji interpolasi yang ada di dalam file.
private static double readInterpolTestFromFile(String path)	Fungsi ini membaca titik uji interpolasi terakhir dari file.
private static Matrix SPLInterpolFile(Matrix A)	Fungsi ini mirip dengan SPLInterpol, tetapi digunakan untuk menangani data interpolasi dari file.

3.2.4. Folder `

3.2.4.1. RegresiBerganda.java

Tabel 3.17. Atribut dari *Class* RegresiBerganda

Nama Atribut	Keterangan
-	Tidak terdapat atribut pada kelas ini

Tabel 3.18. Metode dari *Class* RegresiBerganda

Nama Metode	Keterangan
public static Matrix multiplyMatrices(Matrix A, Matrix B)	Fungsi ini melakukan perkalian dua matriks A dan B.
public static Matrix regresiLinear(Matrix M)	Fungsi ini melakukan regresi linear berganda pada matriks M (data).
public static void	Fungsi ini digunakan untuk

Nama Metode	Keterangan
estimateY(Matrix koefisien, Matrix XP, String output)	memperkirakan nilai Y berdasarkan koefisien regresi yang sudah dihitung sebelumnya dan data prediksi XP.
public static Matrix regresiKuadratik(Matrix M)	Fungsi ini melakukan regresi kuadratik (non-linear) pada matriks M.
public static void printRegresiKuadrat(Matrix koefisien, Matrix XP, String output)	printRegresiKuadrat(Matrix koefisien, Matrix XP, String output):

3.2.5. Folder bicubic

3.2.5.1. BicubicSpline.java

Tabel 3.19 Atribut dari *Class* BicubicSpline

Nama Atribut	Keterangan
-	Tidak terdapat atribut pada kelas ini

Tabel 3.20 Metode dari *Class* BicubicSpline

Nama Metode	Keterangan
public static Matrix fungsiF()	Fungsi untuk menghasilkan matriks 4x16 yang setiap elemennya bernilai f(x,y)/a menyesuaikan rumus $\sum_{i=0}^{3}\sum_{j=0}^{3} x^i y^j$ dengan nilai f(x,y) per baris berturut turut

	adalah f(0,0), f(1,0), f(0,1), f(1,1).
public static Matrix fungsiFx()	Fungsi untuk menghasilkan matriks 4x16 yang setiap elemennya bernilai fx(x,y)/a menyesuaikan rumus $\sum_{i=0}^{3}\sum_{j=0}^{3} ix^{i-1}y^{j}$ dengan nilai fx(x,y) per baris berturut turut adalah fx(0,0), fx(1,0), fx(0,1), fx(1,1).
public static Matrix fungsiFy()	Fungsi untuk menghasilkan matriks 4x16 yang setiap elemennya bernilai fy(x,y)/a menyesuaikan rumus $\sum_{i=0}^{3}\sum_{j=0}^{3} jx^{i}y^{j-1}$ dengan nilai fy(x,y) per baris berturut turut adalah fy(0,0), fy(1,0), fy(0,1), fy(1,1).
public static Matrix fungsiFxy()	Fungsi untuk menghasilkan matriks 4x16 yang setiap elemennya bernilai fxy(x,y)/a menyesuaikan rumus $\sum_{i=0}^{3}\sum_{j=0}^{3} ijx^{i-1}y^{j-1}$ dengan nilai fxy(x,y) per baris berturut turut adalah fxy(0,0), fxy(1,0), fxy(0,1), fxy(1,1).
public static Matrix matrixX()	Fungsi untuk menghasilkan matriks X berukuran 16x16 dengan menggabungkan matriks fungsi F, matriks fungsi Fx, matriks fungsi Fy, dan matriks fungsi Fxy.
public static void interpolasiBicubics(Matrix M)	Prosedur untuk menghitung nilai f(a,b) menyesuaikan rumus bicubicS

3.2.6. Main.java

Tabel 3.21. Atribut dari *Class* RegresiBerganda

Nama Atribut	Keterangan
-	Tidak terdapat atribut pada kelas ini

Tabel 3.22. Metode dari *Class* RegresiBerganda

Nama Metode	Keterangan
public static void main(String[] args)	Prosedur utama program
private static void displayWelcome()	Prosedur utama menampilkan header saat pertama masuk program.
private static void displayExit()	Prosedur utama menampilkan exit saat keluar program.
private static void displayMenu()	Prosedur untuk menampilkan seluruh meu
private static void handleSPL(Scanner userInput)	Prosedur untuk menu spl
private static void handleDeterminan(Scanner userInput)	Prosedur untuk memanggil fungsi determinan
private static void handleInvers(Scanner userInput)	Prosedur untuk memanggil fungsi balikan matriks
private static void handleInterpol(Scanner userInput)	Prosedur untuk memanggil interpolasi
private static void	Prosedur untuk memanggil bicubic

Nama Metode	Keterangan
handleBicubic(Scanner userInput)	
private static void handleRegresi(Scanner userInput)	Prosedur untuk memanggil regresi
public static void printMatrixOutput(Matrix matrix, StringBuilder output)	Prosedur untuk menampilkan matriks
private static int getValidInput (Scanner userInput, int min, int max)	Fungsi untuk mendapatkan input yang valid
private static String getPath(Scanner userInput)	Fungsi untuk menghasilkan path dari nama file yang diinput

BAB IV
EKSPERIMEN

1.a. Solusi SPL Ax = b

a. Input:

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

b. Output

```
Masukkan elemen matriks:
1 1 -1 -1 1
2 5 -7 -5 -2
2 -1 1 3 4
5 2 -4 2 6
SPL tidak memiliki solusi
Simpan hasil ke dalam file (y/n)?
y
Mau disimpan dengan nama file apa?
1a.output
Berhasil menyimpan hasil.
```

<table>
<tr><th>Metode</th><th>Output</th></tr>
<tr><td>Gauss</td><td>Pilih metode penyelesaian:
1. Gauss
2. Gauss Jordan
3. Cramer
4. Matriks balikan
Masukkan pilihan (1-4): 1

Pilih metode input:
1. Input Keyboard
2. Input dari file
Masukkan pilihan (1-2): 2
Masukkan nama file: 1a_SPL
test/Input/1a_SPL
File ditemukan: test/Input/1a_SPL
SPL tidak memiliki solusi
Simpan hasil ke dalam file (y/n)?</td></tr>
<tr><td>Gauss-Jordan</td><td></td></tr>
</table>

Cramer	Pilih metode penyelesaian: 1. Gauss 2. Gauss Jordan 3. Cramer 4. Matriks balikan Masukkan pilihan (1-4): 3 Pilih metode input: 1. Input Keyboard 2. Input dari file Masukkan pilihan (1-2): 2 Masukkan nama file: 1a_SPL test/Input/1a_SPL File ditemukan: test/Input/1a_SPL SPL tidak memiliki solusi
Balikan Matriks	Pilih metode penyelesaian: 1. Gauss 2. Gauss Jordan 3. Cramer 4. Matriks balikan Masukkan pilihan (1-4): 4 Pilih metode input: 1. Input Keyboard 2. Input dari file Masukkan pilihan (1-2): 1 Masukkan jumlah baris: 4 Masukkan jumlah kolom: 5 Masukkan elemen matriks: 1 1 -1 -1 1 2 5 -7 -5 -2 2 -1 1 3 4 5 2 -4 2 6 1.3333333333333321 -2.333333333333332 -1.0 1.0 Simpan hasil ke dalam file (y/n)? n

2. SPL Augmented

a. Input:

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

b. Output

Metode	Output
Gauss	Pilih metode penyelesaian: 1. Gauss 2. Gauss Jordan 3. Cramer 4. Matriks balikan Masukkan pilihan (1-4): 1 Pilih metode input: 1. Input Keyboard 2. Input dari file Masukkan pilihan (1-2): 1 Masukkan jumlah baris: 4 Masukkan jumlah kolom: 5 Masukkan elemen matriks: 1 -1 2 -1 -1 2 1 -2 -2 -2 -1 2 -4 1 1 3 0 0 -3 -3 x1 = -1,0000 x2 = 0,0000 x3 = t3 (variabel bebas) x4 = t4 (variabel bebas) SPL memiliki tak hingga solusi

Gauss Jordan	Pilih metode input: 1. Input Keyboard 2. Input dari file Masukkan pilihan (1-2): 1 Masukkan jumlah baris: 4 Masukkan jumlah kolom: 5 Masukkan elemen matriks: 1 -1 2 -1 -1 2 1 -2 -2 -2 -1 2 -4 1 1 3 0 0 -3 -3 x1 = -1,0000 x2 = 0,0000 x3 = t3 (variabel bebas) x4 = t4 (variabel bebas) SPL memiliki tak hingga solusi
Cramer	
BalikanMatriks	

3. SPL berbentuk persamaan
   a. Input

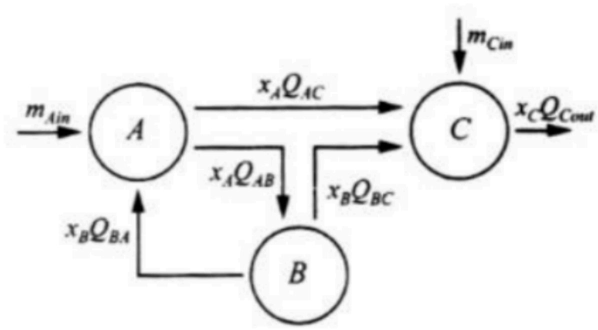
$$
\begin{aligned}
8x_1 + x_2 + 3x_3 + 2x_4 &= 0 \\
2x_1 + 9x_2 - x_3 - 2x_4 &= 1 \\
x_1 + 3x_2 + 2x_3 - x_4 &= 2 \\
x_1 + 6x_3 + 4x_4 &= 3
\end{aligned}
$$

   b. Output

Metode	Output

Gauss	Pilih metode penyelesaian: 1. Gauss 2. Gauss Jordan 3. Cramer 4. Matriks balikan Masukkan pilihan (1-4): 1 Pilih metode input: 1. Input Keyboard 2. Input dari file Masukkan pilihan (1-2): 1 Masukkan jumlah baris: 4 Masukkan jumlah kolom: 4 Masukkan elemen matriks: 8 1 3 2 0 2 9 -1 -2 1 1 3 2 -1 2 1 0 6 4 3 SPL tidak memiliki solusi
Gauss Jordan	Pilih metode input: 1. Input Keyboard 2. Input dari file Masukkan pilihan (1-2): 1 Masukkan jumlah baris: 4 Masukkan jumlah kolom: 5 Masukkan elemen matriks: 1 -1 2 -1 -1 2 1 -2 -2 -2 -1 2 -4 1 1 3 0 0 -3 -3 x1 = -1,0000 x2 = 0,0000 x3 = t3 (variabel bebas) x4 = t4 (variabel bebas) SPL memiliki tak hingga solusi
Cramer	
BalikanMatriks	

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:

$$
\begin{aligned}
\text{A:} \quad & m_{A_{in}} + Q_{BA}x_B - Q_{AB}x_A - Q_{AC}x_A = 0 \\
\text{B:} \quad & Q_{AB}x_A - Q_{BA}x_B - Q_{BC}x_B = 0 \\
\text{C:} \quad & m_{C_{in}} + Q_{AC}x_A + Q_{BC}x_B - Q_{C_{out}}x_C = 0
\end{aligned}
$$

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*.

test > input > 4_SPL

```
120 -60 0 1300
40 -80 0 0
80 20 -150 -200
```

Metode	Output

Gauss	Pilih metode penyelesaian: 1. Gauss 2. Gauss Jordan 3. Cramer 4. Matriks balikan Masukkan pilihan (1-4): 1 Pilih metode input: 1. Input Keyboard 2. Input dari file Masukkan pilihan (1-2): 2 Masukkan nama file: 4_SPL test/Input/4_SPL File ditemukan: test/Input/4_SPL x1 = 14,4444 x2 = 7,2222 x3 = 10,0000 SPL ini memiliki solusi unik
Gauss Jordan	Pilih metode penyelesaian: 1. Gauss 2. Gauss Jordan 3. Cramer 4. Matriks balikan Masukkan pilihan (1-4): 2 Pilih metode input: 1. Input Keyboard 2. Input dari file Masukkan pilihan (1-2): 2 Masukkan nama file: 4_SPL test/Input/4_SPL File ditemukan: test/Input/4_SPL x1 = 14,4444 x2 = 7,2222 x3 = 10,0000 SPL ini memiliki solusi unik

<table>
<tr><td>Cramer</td><td>

```
Pilih metode penyelesaian:
1. Gauss
2. Gauss Jordan
3. Cramer
4. Matriks balikan
Masukkan pilihan (1-4): 3

Pilih metode input:
1. Input Keyboard
2. Input dari file
Masukkan pilihan (1-2): 2
Masukkan nama file: 4_SPL
test/Input/4_SPL
File ditemukan: test/Input/4_SPL
14.444444444444445

7.222222222222222

10.0
```

</td></tr>
<tr><td>BalikanMatriks</td><td>

```
Masukkan pilihan (1-4): 4

Pilih metode input:
1. Input Keyboard
2. Input dari file
Masukkan pilihan (1-2): 2
Masukkan nama file: 4_SPL
test/Input/4_SPL
File ditemukan: test/Input/4_SPL
120.0
-60.0
0.0
1300.0

40.0
-80.0
0.0
0.0

80.0
20.0
-150.0
-200.0
```

</td></tr>
</table>

5. 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:

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

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

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

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

a. 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:

Tanggal (desimal) = bulan + (tanggal / jumlah hari pada bulan tersebut)

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

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:

a. 16/07/2022
b. 10/08/2022
c. 05/09/2022
d. Masukan user lainnya berupa **tanggal (desimal) yang sudah diolah** dengan asumsi prediksi selalu dilakukan untuk tahun 2022.

b. 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.

6.

a. Input:

```
test > input > 6.txt
72.4 76.3 29.18 0.9
41.6 70.3 29.35 0.91
34.3 77.1 29.24 0.96
35.1 68.0 29.27 0.89
10.7 79.0 29.78 1.00
12.9 67.4 29.39 1.10
8.3 66.8 29.69 1.15
20.1 76.9 29.48 1.03
72.2 77.7 29.09 0.77
24.0 67.7 29.60 1.07
23.2 76.8 29.38 1.07
47.4 86.6 29.35 0.94
31.5 76.9 29.63 1.10
10.6 86.3 29.56 1.10
11.2 86.0 29.48 1.10
73.3 76.3 29.40 0.91
75.4 77.9 29.28 0.87
96.6 78.7 29.29 0.78
107.4 86.8 29.03 0.82
54.9 70.9 29.37 0.95
```

b. Output (Dengan regresi linear)

```
Masukkan koefisien yang akan diprediksa:
Masukkan peubah x1: 50,0 76,0 29,30
Masukkan peubah x2: Masukkan peubah x3:
Persamaan regresi linear adalah: f(x) = -3,5078 + (-0,0026{x1}) + (0,0008{x2}) + (0,1542{x3})
Predikisi nilai Y adalah: 0,9384
```

c. (Dengan regresi kuadrat)

```
Persamaan regresi linear adalah: y = -1146,4372 + (0,1840{x0}) + (0,8386{x
1}) + (75,4503{x2}) + (-0,0000{x0^2}) + (-0,0002{x1^2}) + (-1,2404{x2^2})
+ (0,0000{x0}{x1}) + (-0,0064{x0}{x2}) + (-0,0272{x1}{x2})
Prediksi nilai Y adalah: 0.9439080468202121
```

7. Interpolasi Bicubis Spline

Input	Output
21 98 125 153 51 101 161 59 0 42 72 210 16 12 81 96 0 0	Pilih menu: Masukkan pilihan (1-8): 6 Masukkan nama file: 7a_Bicubic test/Input/7a_Bicubic File ditemukan: test/Input/7a_Bicubic 0.0 Hasil interpolasi bicubic: 21,0000
test > input > 7b_Bicubic 21 98 125 153 51 101 161 59 0 42 72 210 16 12 81 96 0.5 0.5	Pilih menu: Masukkan pilihan (1-8): 6 Masukkan nama file: 7b_Bicubic test/Input/7b_Bicubic File ditemukan: test/Input/7b_Bicubic 0.5 0.5 Hasil interpolasi bicubic: 87,7969
test > input > 7c_Bicubic 21 98 125 153 51 101 161 59 0 42 72 210 16 12 81 96 0.25 0.75	Masukkan nama file: 7c_Bicubic test/Input/7c_Bicubic File ditemukan: test/Input/7c_Bicubic 0.25 0.75 Hasil interpolasi bicubic: 117,7322
test > input > 7d_Bicubic 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 0.5 0.5	Masukkan nama file: 7d_Bicubic test/Input/7d_Bicubic File ditemukan: test/Input/7d_Bicubic 0.5 0.5 Hasil interpolasi bicubic: 2,1250

BAB V
KESIMPULAN

5.1. Kesimpulan

Operasi matriks, seperti perhitungan determinan, penemuan matriks balikan, dan proses OBE memiliki peran penting dalam penyelesaian sistem persamaan linear (SPL). Dalam menyelesaikan sistem persamaan linier, terdapat empat metode yang dapat kita digunakan, yaitu metode Gauss, metode Gauss-Jordan, metode matriks balikan, dan kaidah cramer. Dalam mencari nilai determinan dari suatu matriks dapat digunakan metode kofaktor dan metode OBE. Dalam mencari matriks balikan, dapat digunakan metode OBE identitas dan metode matriks adjoin.

Penggunaan SPL meluas ke berbagai aplikasi matematika, seperti interpolasi polinom, interpolasi bicubic spline, dan regresi linear berganda. Interpolasi polinom memungkinkan penentuan persamaan polinom dari sekumpulan titik data yang dikenal, yang dapat digunakan untuk memprediksi nilai di luar data yang diketahui. Sementara interpolasi bicubic spline sangat efektif dalam memperkirakan nilai di antara titik-titik data yang ada dengan mempertimbangkan nilai sekitarnya, terutama berguna dalam pemrosesan citra dan simulasi. Regresi linear berganda, di sisi lain, membantu memprediksi nilai dari satu variabel berdasarkan beberapa variabel lain, seperti dalam prediksi konsentrasi gas berdasarkan faktor-faktor lingkungan.

Secara keseluruhan, berbagai metode penyelesaian SPL dan aplikasinya memberikan fleksibilitas yang luar biasa dalam menyelesaikan masalah di berbagai bidang, seperti matematika, statistik, pemrosesan citra, dan analisis lingkungan. Masing-masing metode memiliki keunggulan dan keterbatasan tersendiri, yang harus disesuaikan dengan jenis masalah yang dihadapi. Kombinasi dari berbagai pendekatan ini memberikan solusi yang lebih akurat dan relevan terhadap masalah yang kompleks di dunia nyata.

5.2. Saran, Komentar, dan Refleksi

Dalam pengerjaan tugas besar ini, kami memerlukan waktu lebih lama untuk beradaptasi dengan bahasa Java, karena bahasa ini relatif baru dan tidak diajarkan secara langsung dalam mata kuliah.

Selain itu, dalam menyelesaikan tugas besar ini, ketelitian sangat diperlukan dalam memahami spesifikasi tugas agar tidak terjadi kesalahan dalam implementasi. Perlu juga dilakukan eksplorasi terhadap berbagai skenario atau kondisi untuk memastikan semua situasi memiliki penyelesaian yang tepat, termasuk dalam menangani kesalahan (error handling).

Melalui tugas besar ini, kami juga belajar mengenai pentingnya memiliki kemampuan untuk bekerja sama, berkoordinasi, dan berkomunikasi dengan baik.

LAMPIRAN

Lampiran 1. Referensi

Howard Anton, Elementary Linear Algebra, 10th edition, John Wiley amnd Sons, 2010

Informatika.stei.itb.ac.id. (2023). Matriks Eselon. Diakses pada 24 Oktober 2023, dari https://informatika.stei.itb.ac.id/~rinaldi.munir/AljabarGeometri/2023-2024/Algeo-02-Matriks-Eselon-2023.pdf

Rowe, D. P. 2018. BiLinear, Bicubic, and In Between Spline Interpolation. Marquette University. https://www.mssc.mu.edu/~daniel/pubs/RoweTalkMSCS_BiCubic.pdf.

Informatika.stei.itb.ac.id. (2023). Sistem Persamaan Linier (SPL): Metode Eliminasi Gauss-Jordan. Institut Teknologi Bandung. Diakses pada 24 Oktober 2023, dari https://informatika.stei.itb.ac.id/~rinaldi.munir/AljabarGeometri/2023-2024/Algeo-03-Sistem-Persamaan-Linier-2023.pdf

Informatika.stei.itb.ac.id. (2023). Determinan (Bagian 2). Diakses pada 24 Oktober 2023, dari https://informatika.stei.itb.ac.id/~rinaldi.munir/AljabarGeometri/2023-2024/Algeo-09-Determina n-bagian2-2023.pdf

Lampiran 2. Tautan Repository

Tautan Prototipe : https://github.com/FaqihMSY/Algeo01-23057/blob/main/src/Main.java